

C3



⑩ BUNDESREPUBLIK
DEUTSCHLAND



DEUTSCHES
PATENT- UND
MARKENAMT

⑫ Übersetzung der
europäischen Patentschrift

⑨ EP 0 573 503 B.1

⑩ DE 692 32 869 T.2

⑮ Int.-Cl. 7
G 06 F 9/44
G 06 F 17/40
G 01 R 13/04
G 01 R 29/00
G 01 R 13/34

② Deutsches Aktenzeichen: 692 32 869.6
③ PCT-Aktenzeichen: PCT/AU92/00076
④ Europäisches Aktenzeichen: 92 905 546.5
⑤ PCT-Veröffentlichungs-Nr.: WO 92/015959
⑥ PCT-Anmeldetag: 25. 2. 1992
⑦ Veröffentlichungstag der PCT-Anmeldung: 17. 9. 1992
⑧ Erstveröffentlichung durch das EPA: 15. 12. 1993
⑨ Veröffentlichungstag der Patenterteilung beim EPA: 11. 12. 2002
⑩ Veröffentlichungstag im Patentblatt: 4. 9. 2003

⑭ Unionspriorität: 486991 28.02.1991 AU
⑮ Patentinhaber: Associative Measurement Pty. Ltd., North Ryde, Neusüdwaies, AU.
⑯ Vertreter: Patent- und Rechtsanwälte Bardehle, Pagenberg, Dost, Altenburg, Geissler, 81679 München
⑰ Benannte Vertragsstaaten: AT, BE, CH, DE, ES, FR, GB, IT, LI, LU, NL

⑱ Erfinder: WILLIAMS, Victor, Donald, Balmain, AU; KEEBLE, Brian, John, Neutral Bay, AU; OATES, David, John, Quakers Hill, AU; CAMROS, Guillermo, Alejandro, North Rocks, AU.

⑳ WISSENSCHAFTLICHES EMULATORGERÄT

DE 692 32 869 T.2

Anmerkung: Innerhalb von neun Monaten nach der Bekanntmachung des Hinweises auf die Erteilung des europäischen Patents kann jedermann beim Europäischen Patentamt gegen das erteilte europäische Patent Einspruch einlegen. Der Einspruch ist schriftlich einzureichen und zu begründen. Er gilt erst als eingelegt, wenn die Einspruchsgebühr entrichtet worden ist (Art. 99 (1) Europäisches Patentübereinkommen).

Die Übersetzung ist gemäß Artikel 153 Abs. 1 IntPatÜG 1991 vom Patentinhaber eingereicht worden. Sie wurde vom Deutschen Patent- und Markenamt inhaltlich nicht geprüft.

DE 692 32 869 T.2

05.03.02

51.2

692 32 869 6-08

Associative Measurement PTY. LTD.

6. März 2003

S.17755 EP/DE AI/PSp/bb

Die vorliegende Erfindung bezieht sich auf wissenschaftliche Instrumente und insbesondere auf einen Emulator, welcher die Funktionen verschiedener wissenschaftlicher Instrumente erlaubt, wenn sie verbunden sind, emuliert oder reproduziert zu werden, ohne eines Kaufes, einer Installation, einer Verbindung und eines Tests der verschiedenen wissenschaftlichen Instrumente zu bedürfen. In ihrer bevorzugten Form erlaubt die vorliegende Erfindung, dass Messungen gemacht werden und Messprozesse verwendet werden, welche nicht möglich waren, gemacht zu werden und verwendet zu werden mit den verbundenen Instrumenten.

HINTERGRUNDTECHNIK

In vielen Handlungsfeldern, die eine wissenschaftliche Grundlage haben, werden wissenschaftliche Instrumente verwendet, um die Ausgaben verschiedener Vorrichtungen zu analysieren, aufzuzeichnen und zu überwachen. Solche Vorrichtungen schließen Spannungsmesser, elektrokardiographische (ECG) Vorrichtungen, Mikrofone und Druck, Temperatur, Durchflussraten und ähnliche Messwandler ein. Entsprechend werden solche wissenschaftlichen Instrumente im Bauingenieurwesen, in der Elektrotechnik, der Akustik, dem hydraulischen Ingenieurwesen, in chemischen Prozessen, im biomedizinischen Ingenieurwesen, usw. verwendet.

Eine große Zahl solcher wissenschaftlicher Instrumente ist im Allgemeinen erforderlich, um die gewünschten Messungen durchzuführen. Solche Instrumente schließen Generatoren ein für verschiedene Wellenformen (wie z.B. Sinus, Quadrat, Rampe und Dreieck), Signalverarbeitungsvorrichtungen, wie z.B. Differenzierer, Integratoren, Filter, Multiplizierer, usw., Analytiker, wie z.B. solche, die

05.02.02

-12-

erforderlich sind, um die schnelle Fourier-Transformation durchzuführen und verschiedene Aufzeichnungsvorrichtungen, wie z.B. ein Diagramm-Aufzeichner, ein Datenscribe, ein Kathodenstrahl-Oszilloskop oder ein Übergangsaufzeichner.

10 Solche Instrumente oder Vorrichtungen sind verhältnismäßig teuer und so besitzt jede Forschungseinrichtung oder ähnliche Organisation nur eine beschränkte Anzahl solcher Vorrichtungen. Dementsprechend gibt es einen erheblichen Wettbewerb unter Personen oder Gruppen innerhalb solcher Organisationen, welche die Vorrichtungen zu benutzen wünschen. Wenn man die Vorrichtungen erhalten hat, um den vorgesehenen Betrieb durchzuführen, ist es notwendig, dass diese Vorrichtungen gemeinsam angeordnet, verbunden und getestet werden, um sicher zu stellen, dass die Verbindungen richtig sind. Nur wenn diese Prozedur durchgeführt worden ist, ist es dann möglich, den vorgesehenen Betrieb aufzunehmen.

15 Nach dem Stand der Technik ist bekannt, Instrumente bereit zu stellen, die im Wesentlichen weitfortgeschrittene Kathodenstrahl-Oszillographen sind. Ein solches Instrument wird unter dem Namen SUPERSCOPE von G. W. Instruments aus Summerville, Ma. USA 02143, verkauft und erlaubt, Wellenformen durch eine im
20 Wesentlichen einen Speicheroszillographen darstellende Vorrichtung aufzunehmen und ebenso auf einem APPLE (registriertes Warenzeichen) Macintosh (registriertes Warenzeichen) Computer anzuzeigen. Diese Vorrichtung scheint jedoch keine Instrumente zu emulieren, sondern sie nimmt lediglich die Ausgabe solcher Instrumente auf und speichert sie zur darauf folgenden Anzeige.

25 Es ist auch bekannt, Software zur Datenerfassung zu verwenden. Ein solches Programm, das von LABTECH aus Wilmington, Ma und San Francisco, Kalifornien, USA, angeboten wird, ist eine graphische Schnittstelle, die wiederum Signale annimmt von verschiedenen Hardware-Elementen, die sich außerhalb des Computers befinden. Die Software sammelt Daten von mehrfachen Kanälen, führt eine
30 Analyse durch und, falls nötig, eine Reduktion der Daten und erzeugt Anzeigen.

Diese Aktivität ist in Echtzeit verfügbar. Es wird wiederum kein Versuch gemacht, die Funktion wissenschaftlicher Instrumente zu emulieren. Ein weiterer Unterschied besteht darin, dass die graphische Schnittstelle eine Anzahl verschiedener Maschinen miteinander verbindet, anstatt im Wesentlichen in eine Einheit integriert zu sein.

Ein Datenakquisitions- und Analysesystem, das unter dem Handelsnamen LABVIEW2 von National Instruments aus Austin, Texas, USA, gehandelt wird, ist ebenso bekannt. In diesem System werden Icons verwendet, um sowohl Datenerfassungsfunktionen als auch Datenanalysefunktionen darzustellen. Diese Icons werden verwendet, um ausführbaren Code zu erzeugen, der lediglich von einem Personalcomputer ausgeführt wird, in welchem die Software des Systems geladen ist. Dieses System verwendet keine Hardware, sondern lediglich Software. Die Datenerfassungs- und Analysefunktionen können sequentiell verbunden werden, so dass eine Dateneingabe in den Computer zuerst erfasst und dann analysiert wird. Schließlich werden die analysierten Daten dargestellt. Es sei bemerkt, dass der Computer eingesetzt wird, um die gewünschten Verbindungen zwischen externen Instrumenten herzustellen, aber dass er keine Instrumente emuliert oder ein Programm kompiliert, um die Signalverarbeitungsfunktionen einer Gruppe von miteinander verbundenen Instrumenten zu reproduzieren. Zum Beispiel wird keine Vorkehrung getroffen zur Rückkopplung von einem Icon zurück zu einem anderen im Sinne einer Steuerung. Die niedrigen Verarbeitungsgeschwindigkeiten und die unbestimmte Natur der Multitasking-Umgebung des APPLE (eingetragenes Warenzeichen) Personalcomputers erfordern die Bereitstellung einer Zeit-"Koordinate" zur Begleitung der erfassten Daten.

Dieses System ist ein virtuelles Instrument, d.h. es ist lediglich durch eine Software-Simulation gekennzeichnet und muss im Gegensatz gesehen werden mit dem neuen Konzept einer Emulation. Emulation umfasst flexible, mehrfach verwendbare, wiederprogrammierbare Hardware, die Code ausführt, der von einem

05.03.02

-4-

graphischen Compiler bei hoher Verarbeitungsgeschwindigkeit abgeleitet ist, der Echtzeit-Instrumentenemulation erlaubt.

- 5: US 4315.315 bezieht sich auf einen Prozess zur automatischen Erzeugung eines Computerprogramms in einer Maschinen-Assembler-Sprache unmittelbar aus einem zweidimensionalen Netzwerk, welches den Datenfluss und die Steuerlogik darstellt, die verlangt wird, um auf einem angegebenen, allgemein-verwendbaren Digitalrechner ausgeführt zu werden. Eine Emulation wird jedoch nicht durchgeführt.

10:

Der Artikel "Using Mathematica in Support of LabView: Power in the Laboratory", NORTHCON Conf. Rec. Seattle, WA, USA, 9-11. Okt. 1990, Seiten 353-358, bezieht sich auf eine Graphik-orientierte Computersprache zum Erzeugen von Programmen, welche auf einem Personalcomputer ausgeführt werden. Es wird keine Emulation durchgeführt.

- 20: Deshalb ist die Aufgabe der vorliegenden Erfindung, einen wissenschaftlichen Instrumenten-Emulator bereit zu stellen und ein entsprechendes Verfahren, welche die Funktion von wissenschaftlichen Instrumenten reproduzieren. Diese Aufgabe wird gelöst durch den Emulator gemäß Anspruch 1 bzw. durch das Verfahren gemäß Anspruch 8. Weitere Ausführungsformen sind in den abhängigen Ansprüchen bestimmt.

- 25: Die Aufgabe des Emulators ist es, mit Hilfe einer Kombination von sowohl Hardware als auch Software die Funktionen verschiedener wissenschaftlicher Instrumente zu reproduzieren, um damit nicht nur individuelle Funktionen von Instrumenten zu erlauben, die reproduziert werden sollen, sondern erlauben auch, dass solche reproduzierten Funktionen miteinander verbunden werden können, so dass der Emulator als Ganzes die gesamte Funktion eines Feldes, das aus miteinander verbundenen wissenschaftlichen Instrumenten gebildet wird, reproduzieren kann.

000000

Eine Anzahl von sehr wesentlichen Vorteilen wird auf diese Weise erreicht. Zu-
erst werden die Kosten der Vorrichtungen wesentlich reduziert, da die einzelnen
wissenschaftlichen Instrumente, deren Funktionen reproduziert werden sollen,
5 nicht angeschafft werden müssen. Zweitens wird auch eine wesentliche Ersparnis
in der Einrichtungszeit erreicht, da es nicht notwendig ist, dass solche Instrumente
von verschiedenen Orten eingekauft und an einem Punkt zusammengebaut wer-
den müssen und physikalisch verbunden werden müssen mit Kabeln, Steckern,
Adaptern, usw. Drittens werden die erzielten Ergebnisse dadurch sehr verbessert,
10 dass die Ausgabe der Gruppe von miteinander verbundenen wissenschaftlichen
Instrumenten nicht nur in Echtzeit angezeigt werden kann, sondern sie zusätzlich
in Echtzeit verarbeitet wird und sie kann ebenso gespeichert werden zur nachfol-
genden Manipulation, Analyse und Bewertung. Schließlich können die Ergebnisse
von gleichzeitigen oder parallelen Messungen leichter aufeinander bezogen wer-
15 den, so dass die Zwischenbeziehungen zwischen Messungen gesichtet werden
können, insbesondere in komplexen Systemen.

Die Erfindung ist dargelegt im angefügten Vorrichtungsanspruch 1 und im Ver-
fahrensanspruch 8.

20 Vorzugsweise sind die Instrumenteneinheiten jeweils dargestellt durch einen ent-
sprechenden Icon, der angezeigt werden kann auf dem Videoanzeigeschirm wäh-
rend des Betriebs des Installationsprogramms, um es dem Operator zu erlauben,
die gewünschte Instrumentationseinheit auszuwählen und in dem Feld zu platzie-
25 ren:

Der Computer erzeugt im Speicher eine Anzahl von verschiedenen Programmen,
welche das Feld von Instrumentationseinheiten darstellen. Diese Programme sind
auf dem Prozessor im Computersystem verteilt, um das gewünschte Feld von In-
30 strumentationseinheiten zu emulieren. Vorzugsweise sind die vom Compiler er-
zeugten Programme im Speicher gespeichert, um vorkonfigurierte Instrumentati-

onsfelder zu erzeugen, welche sofort in der Lage sind, gewünschte Verarbeitungsfunktionen auszuführen. Diese Programme speichern das akkumulierte intellektuelle Werk des Benutzers.

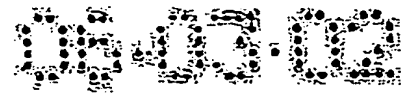
5 Vorzugweise ist die Anzahl von Prozessoren, Videogeneratoren und Analogsignalmodulen erweiterbar, um den gewünschten Grad von Komplexität und/oder Verwendbarkeit des Feldes von Instrumentationseinheiten zu erreichen.

10 Dem Analogsignalmodul steht ein elektrisches Echtzeitsignal zur Verfügung, welches verwendet werden kann, um andere Hardwareelemente zu betreiben oder anzusteuern. Vorzugweise stellt es sowohl einen analogen als auch digitalen Ausgang bereit.

15 Die Ein-/Ausgabe des wissenschaftlichen Instrumentenemulators ist vorzugsweise auch zugänglich von anderen Ressourcen, welche innerhalb des Computers verfügbar sind, einschließlich Netzwerk-Kommunikationsschnittstellen (RS232, ETHERNET, etc.) und Busschnittstellen, wie z.B. IEEE-488-GPIB, ISA und EISA. Diese Ressourcen können unabhängig bereit gestellt werden durch den Käufer oder OEM's (original equipment manufacturers).

20 Die Daten, welche in dem Speichermittel gespeichert sind, stehen vorzugsweise auch für den Daten-Export zur Verfügung zu verschiedenen Standard-Computerpaketen, wie denjenigen, die unter den Handelsmarken EXCEL, LOTUS und AXUM verkauft werden, wobei die angesammelten Daten manipuliert werden können für eine nachfolgende graphische Darstellung und zur Tabellierung, um die Berichterzeugung zu vereinfachen.

25 Gemäß eines zweiten Gesichtspunkts der vorliegenden Erfindung wird ein wissenschaftliches Instrument offenbart zum Messen und Aufzeichnen elektrischer Wellenformen, wobei das Instrument einen Computer umfasst, der eine Zentraleinheit und elektronische Speichermittel aufweist, welche in einem Gehäuse



untergebracht sind, wobei das Gehäuse die Bereitstellung von mindestens einem Floppy-Disk-Schacht umfasst, und eine Vielzahl elektrischer Verbinder, die untergebracht sind in dem Raum, der vorgesehen ist für einen Floppy-Disk-Schacht, wobei die elektrischen Verbinder verbunden sind mit dem Speichermedium.

5

Gemäß eines dritten Gesichtspunktes der vorliegenden Erfindung wird ein Kompilerverfahren offenbart zum Erzeugen von Objektcode, um die mathematische/Signalverarbeitungs-Prozedur eines elektrischen Schaltkreisfunktionsblocks zu implementieren, der mindestens eine Eingabe aufweist, um daraus eine Ausgabe zu bilden, wobei das Verfahren die Schritte umfasst der Darstellung der Funktion als eine Folge elementarer mathematischer Schritte, von denen jeder selbst unmittelbar darstellbar ist in dem Objektcode, und Anordnen der Objektschritte in eine Folge zur sequentiellen Ausführung, beginnend mit der Eingabe/ den Eingaben. Vorzugsweise kann ein Parameter des Funktionsblocks spezifiziert werden. Ebenso offenbart ist ein Kompilierungsverfahren zum Erzeugen von Objektcode, um die mathematischen/Signalverarbeitungsfunktion eines elektrischen Schaltkreises zu implementieren, welche mindestens eine Eingabe und mindestens eine Ausgabe aufweist und gebildet wird durch Verbindung einer Vielzahl von funktionalen Blöcken, von denen jeder eine mathematische/Signalverarbeitungsfunktion aufweist, für welche ein Objektcode kompiliert wurde in Übereinstimmung mit dem Obigen, wobei das Verfahren die Schritte aufweist der Darstellung der elektrischen Schaltkreis-mathematischen/Signalverarbeitungsfunktion als Folge von Ereignissen, von denen jedes darstellbar ist in dem Objektcode, und Anordnen der Objektschritte in einer Folge zur sequentiellen Ausführung, beginnend mit der Eingabe/ den Eingaben.

15
20
25

BESCHREIBUNG DER ZEICHNUNGEN

Eine Ausführungsform der vorliegenden Erfindung wird nun beschrieben werden, unter Bezugnahme auf die Zeichnungen, in welchen:

30

09-13-82

-8-

Fig. 1 ein schematisches Blockdiagramm der Hardwaremodifikationen ist, die an einem IBM-PC erforderlich sind,

Fig. 2 ein Blockdiagramm des Analogmoduls von Fig. 1 ist,

Fig. 3 ein Schaltkreisdigramm von einem von identischen Relais von Fig. 2 ist,

Fig. 4 eine schematische Karte der Speicheranordnung innerhalb des Computers ist,

Fig. 5 ein Blockdiagramm der Echtzeit-Video-gedruckten-Schaltung von Fig. 4 ist,

Fig. 6 eine Bildschirmanzeigenaufstellung der Icons ist, die verschiedene Instrumentationseinheiten darstellen, die innerhalb des Bibliotheksprogramms gespeichert sind,

Fig. 7 ein typisches Feld ist, welches durch Verbindung der verschiedenen Instrumentationseinheiten gebildet wird,

Fig. 8 ein Blockdiagramm eines vergleichsweise einfachen Feldes ist, das nützlich ist in biomedizinischen Anwendungen,

Fig. 9 eine Wiedergabe der Bildschirmanzeigefenster ist, die dem Feld von Fig. 8 entsprechen,

Fig. 10 ein Feld ist, das eine lineare Interpolation bildet,

Fig. 11 die Ausgabe der drei Bildschirmanzeigen zeigt, die in Fig. 10 veranschaulicht sind.

Fig. 12 ein Feld ist, welches eine phasenverriegelte Schleife bildet,

Fig. 13 die Ausgabe von den drei Bildschirmanzeigen, veranschaulicht in Fig. 12, zeigt.

Fig. 14 ein Feld ist, welches eine analoge Lösung bereit stellt für eine Differenzialgleichung zweiter Ordnung.

Fig. 15 die Ausgabe der beiden Bildschirmanzeigen, veranschaulicht in Fig. 14, zeigt und.

Fig. 16 ein komplexeres Feld ist.

Die Anhänge I-IV listen verschiedenen Programmfragmente auf, die nachfolgend beschrieben werden.

BESCHREIBUNG DER BEVORZUGTEN AUSFÜHRUNGSFORM

Es sei jetzt auf Fig. 1 Bezug genommen, worin die Vorrichtung der bevorzugten Ausführungsform ganz enthalten sein kann innerhalb des Gehäuses 1 eines herkömmlichen IBM (eingetragenes Warenzeichen) Personal Computers, der einen ISA oder EISA Bus aufweist auf der Basis des ursprünglichen IBM-AT. Innerhalb des Gehäuses 1 sind die üblichen Komponenten untergebracht von einer Zentraleinheit (CPU) 2, einem Speicher 3 und einem 3MHz-Bus 4.

Innerhalb eines Reserve 5 1/4-Zoll-Floppy-Disk-Einschubs ist ein Analogmodul 6 untergebracht, auf welchem bestimmte Eingabestecker 7, bestimmte Ausgabestecker 8 und allgemeine Ein-/Ausgabeleitungen 9 für Verstärker, Frequenzzähler, Abtastaktsynchronisierung, digitale Eingaben und Ähnliches montiert sind.

Auf dem Bus 4 befinden sich 4 Schlitze für gedruckte Schaltkreiskarten 9. Die 4 PC-Karten 9 sind bezeichnet mit A, B, C bzw. D. Die drei benachbarten PCBs A bis C einschließlich stellen jeweils einen Master-PCB, einen Slave-PCB und einen Video-PCB dar. Der Video-PCB treibt seinerseits eine bekannte VGA-gedruckte Schaltkreiskarte D, die von Tseng Labs verkauft wird, welche eine 800 x 600 Auflösung in 256 Farben bereit stellen kann. Diese PC-Karte D wird direkt verbunden mit dem Videoanzeigeschirm 10.

Innerhalb des Analogmoduls 6 befinden sich folgende Systemressourcen:

10

1 VIER isolierte (optionale) analoge i/p-Kanäle. Jeder Kanal weist eine programmierbare 9-120dB-Verstärkungsleistung auf (3 Mikrovolt Auflösung 0 Signal zu Rauschverhältnis von eins), programmierbare Anti-Alias-Filtrierung und eine ADC-Umwandlung mit 12-Bit-Auflösung. Jeder Kanal kann AC- oder DC-gekoppelt sein mit langen AC-Kopplungszeitkonstanten (2 Minuten) und weist unabhängige Steuerungen für AC- oder DC-Offsets auf, die gesteuert werden können von den Laufzeitbildschirmen. Die Abtastrate kann 15 KHz pro Kanal (abhängig von der Projektverarbeitungslast) betragen und die Zahl der Analogmodule, die mit der gleichen Slave-Prozessorkarte verbunden sind. Die Eingaben sind auf 3-5 KV Gleichstrom isoliert.

20

2 ZWEI Analogausgaben mit einem Spannungsbereich von +/- 10 V und einer Stromstärke von +/- 100 mA. Diese können verwendet werden zur Vorspannung von Spannungsmessern (AC- oder DC-getrieben), Steuerausgaben, etc.

25

3 VIER auswählbare analoge Ausgaben von hohem Niveau, eine von jedem der obigen Verstärker. Diese treiben digitale FM-Band-Recorder an, um selten auftretende Ereignisse zu speichern zur Wiedergabe in dem Prozessor (2).

30

ÖDER

02.03.02

-11-

VIER auswählbare Eingaben von hohem Niveau zu jedem der obigen Verstärkerkanäle. Das System wird in diesem Modus geschaltet zum Wiedergeben von Ereignissen, welche im Ausgabemodus auf Band genommen wurden.

- 3 4. ACHT Bits einer Masse-referenzierten digitalen Eingabe.
- 5 5. ACHT Bits einer digitalen Ausgabe, die verwendet werden kann für Relais-Treiber oder Ereignisindikatoren.
- 10 6. EINE Frequenzgeneratorausgabe (Taktgenerator) 0-2 MHz 0,1% Genauigkeit.
- 7 7. EIN Ereigniszähler/Frequenzzähler Eingabe 0,1 Hz - 8MHz.
- 15 8. EINE 5-Volt Referenz 100 mA \pm 5% (für Spannungsmesser, etc.).
- 9 9. EINE Abtasttaktausgabe Referenzleitung zum Synchronisieren einer Abtastung zwischen mehreren wissenschaftlichen Instrumentenemulatoren der bevorzugten Ausführungsform.
- 20 10. EINE Abtastakteingabe Referenzleitung zum Synchronisieren einer Abtastung von einem "Master"-wissenschaftlichen Instrumentemulator (zur Verwendung mit "Slave"-Emulatoren), und.
- 25 11. FÜNF Massekabel.

Das Analogmodul 6 und die PC Karten 9 sind alle durch verschiedene Hilfsbusse 11, 12, 13 bzw. 14 verbunden.

- 30 Wie in Fig. 2 zu sehen, wird das Analogmodul 6 von Fig. 1 bereit gestellt mit vier analogen Ein-/Ausgabeverbindern 20, vier analogen Eingängen 21, zwei analogen

Ausgaben 22, einem Frequenzausgabezehlereingang 23, einem Takt Ausgang 24, einem 8-Bit digitalen Eingang 25, einem 8-Bit digitalen Ausgang 26, einer 5 Volt Referenzspannung 27 und einem Slave-Synchronisierausgang 28.

5 Jeder der analogen Eingänge 21 ist verbunden über einen Frontendverstärker 31 mit einem Isolator 32, dessen Ausgang verbunden ist mit einem Relais 33. Das Relais 33 ist auch mit den analogen Ein-/Ausgabeverbindern 20 und einem Verstärker 34 verbunden, welcher eine programmierbare Verstärkungsleistung, eine AC/DC-Kopplung aufweist und einen AC-Comer und einen DC-Offset aufweist.
10 Die Ausgabe des Verstärkers 34 wird ihrerseits zu einem Abtast- und Halteschaltkreis 35 geleitet, dessen Ausgabe empfangen wird von einem analogen Multiplexer 36. Die Ausgabe des Multiplexers 36 wird über einen A/D-Wandler 37 zu dem Hilfsbus 11 geleitet, der die analogen Module 6 und die PC-Karte 9B verbindet.

15 Der Betrieb der Verstärker 34 und der Abtast- und Halteschaltkreise 35 wird gesteuert von einer digitalen Steuereinheit, einem Adress-Dekodierer und einem A/D-D/A-Sequenzierer 39, welcher sowohl Daten vom Bus 11 als auch Abtasttakt- und Sequenzierertaktsignale empfängt. Die Steuereinheit/Dekodierer/Sequenzierer 39 gibt auch über D/A-Wandler 40 zu den analogen
20 Ausgängen 22 über einen Ausgabeverstärker 41 aus.

Der Frequenzzehlereingang 23 bzw. Takt Ausgang 24 kommunizieren direkt mit einem Zähler 42, der seinerseits direkt mit dem Hilfsbus 11 kommuniziert.

25 Jeder der digitalen Eingänge 25, der digitalen Ausgänge 26, Referenzspannung 27 und des Slave-Synchronisierausgangs 28 ist verbunden mit einem digitalen Ein-/Ausgabeschaltkreis 43, der seinerseits direkt mit dem Hilfsbus 11 verbunden ist.

30 Die Taktanordnungen des Schaltkreises, der in Fig. 2 veranschaulicht ist, sind in zwei Sequenzen geteilt. Die erste Sequenz betrifft die digitale Eingabe und Aus-

gabe. Wenn es vom Programm verlangt wird, wird diese digitale Eingabe und Ausgabe in Gang gesetzt durch individuelle Befehle von einer im Wesentlichen herkömmlichen Datenerfassungssteuereinheit, die einen Teil des Slave-Prozessors auf PCB-9B (Fig. 1 und Fig. 4) bildet.

Die zweite Sequenz ist der Fluss digitaler Daten, die umgewandelt wurden von analogen Eingaben oder die umgewandelt werden sollen, um analoge Ausgaben bereit zu stellen. Diese digitalen Daten werden empfangen und abgeschickt unter der Steuerung der Steuereinheit/Dekodierer/Sequenzierer 39, die voreingestellt werden kann, um die erforderliche Zahl ankommender oder abgehender analoger Kanäle zu betreiben. Der Controller/Dekodierer/Sequenzierer 39 führt einen vollständigen Zyklus des Eingebens und Ausgebens oder einer Sequenz in jeder Abtastperiode aus und er führt dies mit minimalem Prozessoraufwand durch und erhöht auf diese Weise die Betriebsgeschwindigkeit des Datenerfassungs-Controllers, auf den oben Bezug genommen wurde auf dem Slave-Prozessor von PCB-9B.

Andere Funktionen des Schaltkreises von Fig. 2, wie z.B. die Frequenz, die ausgegeben werden soll als die Takt Ausgabe 24, der Bereich der Frequenz, der gezählt werden soll von Frequenzzählereingang 23, und irgendein Synchronisations-signal, welches erforderlich ist für das Slave-Synchronisations-signal 28, werden eingestellt zu Beginn der Ausführung des graphischen Compiler-Programms durch geeignetes Angeben des entsprechenden Icons.

Fig. 3 veranschaulicht im Detail die Natur des Relais 33, das zwischen zwei Positionen geschaltet werden kann. In der gezeigten Position wird die Eingabe vom analogen Eingang/Ausgang 20 zum Verstärker 34 geführt, womit die analogen Eingänge 21 isoliert sind. In der alternativen Stellung wird der Eingang von den analogen Eingängen 21 zum Verstärker 34 geführt, aber er wird auch als Ausgang verfügbar gemacht am analogen Eingang/Ausgang 20.

Fig. 4 ist eine bildhafte Darstellung der Prozessorarchitektur, die eine Speicherkarte beinhaltet; der Host-Computer 22 mit seinem zugeordneten Speicher 3 ist verbunden mit dem Video-PCB 9C, welches seinerseits mit dem Slave-PCB 9B verbunden ist.

Wie angegeben in Fig. 4, können bis zu 8 analoge Module 6 verbunden werden zu jeder PCB 9B und bis zu 4 Slave-PCBs 9B können hinzugefügt werden.

Die Speicherkarte ist im Wesentlichen dreidimensional, wobei der Speicher 3 des Host-Computers 2 überlappt und damit zugänglich ist mit den anderen Abschnitten des Schaltkreises, die den Speicher verwenden.

Ein Blockdiagramm des Video-PCB 9C ist veranschaulicht in Fig. 5. In diesem Diagramm sind die Verbindungen zwischen dem herkömmlichen Videografikadapter (VGA) des Host-Computers hergestellt über den herkömmlichen VGA-Verbin-
der 50. Die Verbindungen zwischen dem Host-Computer 2 und dem Video-PCB 9C werden über Bus 4 geführt. Ebenso werden die Verbindungen zwischen dem Video PCB 9C und jedem der Slave-PCBs 9B über Bus 13 geführt, wie oben angegeben in Fig. 1.

Daten, die entweder vom Bus 4 und/oder 13 empfangen werden, werden über einen logischen Dual-Port-Schaltkreis 51 zu einem Zuerst-ein-zuerst-aus (FIFO)-Puffer 52 geführt. Der FIFO 52 gibt an einen logischen Feldschaltkreis 53 aus, der drei getrennte wahlfreie Speicher aufweist: einen horizontalen RAM 54, einen vertikalen RAM 55 und einen statischen RAM 56.

Der Ausgang des logischen Felds 53 wird über den Vergleichler 57 zu einem Videogenerator 58 geführt und dann zu dem VGA-Verbin-
der 50. Zusätzlich empfängt das logische Feld 53 auch drei Signale von dem VGA-Verbin-
der in Form von horizontalen Synchronisationspulsen, vertikalen Synchronisationspulsen und einem Punkt-Takt.

Im Wesentlichen nimmt das logische Feld 53 die Daten, die über Busse 4 und/oder 13 geliefert werden, auf und berechnet Pixel, um die einzelnen Pixel zu ersetzen, die von dem Videografikadapter (VGA) 12 erzeugt werden, und es stimmt mit Multimedia-Standards überein zur Kommunikation des Videobildes auf Bus 14.

Der Speicher 3 (Fig. 1 und 4) weist eine Bibliothek von Instrumentationseinheiten auf. Davon wird jede dargestellt durch einen Icon, und Fig. 6 liefert einen Eindruck des Bereichs von Instrumentationseinheiten, die ausgewählt werden können von einer gegebenen Bibliothek. Durch Verwendung einer Maus in bekannter Art kann der Operator ausgewählte Instrumentationseinheiten verbinden aus der Icon-Liste von Fig. 6, um ein Feld zu bilden von miteinander verbundenen Instrumentationseinheiten, veranschaulicht in Fig. 7. Während der Erzeugung des Felds von Fig. 7 überprüft die Maschine unter der Steuerung von Software, dass das Feld keine nicht verbundenen Eingaben, logisch nicht akzeptierbare Verbindungen und ähnliche Defekte aufweist. Jeder detektierte Effekt wird angezeigt.

Nachdem das Feld verbunden wurde zur Zufriedenheit sowohl des Operators als auch des Setup-Programms, das während dieser Phase verwendet wird, wird dann ein Compiler-Programm zum Ablauf gebracht, welches aus der graphischen Darstellung des Feldes ausführbaren Objektcode erzeugt, der die gesamte Signalverarbeitungsfunktion des gesamten Feldes ausführt. Als Folge davon, wenn in Echtzeit das Eingangssignal angewandt wird auf das Feld, wird/werden das eingehende Signal/die eingehenden Signale manipuliert und der eine oder mehrere Ausgänge des Feldes werden angezeigt in Echtzeit auf den Videofenstern, die angezeigt werden können auf dem Schirm 10, gespeichert auf Platte, usw.

Wesentlich für das Verständnis der Art, in welcher das Compiler-Programm arbeitet, ist eine Würdigung, dass jeder Icon selbst ein Minifeld darstellt, das aufgebaut werden kann von sehr fundamentalen Schritten, die ihrerseits leicht von dem

Programm ausgeführt werden können. Wenn z.B. die Grundschrirte angenommen werden, Addition und Subtraktion zu sein, dann kann eine Modifikation betrachtet werden als wiederholte Addition und eine Division kann betrachtet werden als wiederholte Subtraktion. Mit diesem Hintergrund vor Augen kann es gewürdigt werden, dass ein Rampenspannungsgenerator erzeugt werden kann von einem Null-Anfangsniveau durch sukzessive Addition von sehr kleinen Inkrementen, bis ein vorbestimmtes Niveau erreicht wird. Dann wird dieses Niveau selbst subtrahiert, um den Null-Startpunkt wieder zu erzeugen. Dann werden die kleinen Additionen wiederum durchgeführt, usw.

Eine Berücksichtigung der verschiedenen Icons, die in Fig. 3 dargestellt sind, wird ebenso auf die Würdigung führen, dass der Icon selbst im Wesentlichen eine graphische Form jeder gewünschten Gestalt darstellt, die entworfen ist, um eine Darstellung einer bestimmten mathematischen oder Signalverarbeitungsfunktion zu überbringen, die sowohl vom Benutzer gesehen werden kann als auch identisch verstanden wird vom graphischen Compiler-Programm. Die Gestalt enthält mindestens einen Eingabeknoten und/oder mindestens einen Ausgabeknoten, die jeweils Eingangs- und Ausgangspunkte für Datenströme darstellen. Zum Beispiel wird ein Addierer, der Eingabeknoten A und B und einen Ausgabeknoten C aufweist, kompiliert, um das Programm $C = A + B$ zu ergeben. Diese Icon-Elemente, welche lediglich einen oder mehrere Ausgangsknoten aufweisen, werden Quell-Icon-Elemente genannt (z.B. ein Spannungsgenerator), während diejenigen Icons, die lediglich einen oder mehrere Eingabeknoten aufweisen, ein Ziel-Icon genannt werden (z.B. ein Anzeigefenster).

Ein verbindender Pfad, der irgendeinen Ausgabeknoten verbindet und in einem Eingabeknoten endet, wird ein Datenstrom genannt. Der Datenstrom trägt den Datentyp, der mit dem Ausgabeknoten assoziiert wird, und als Folge muss der Eingabeknoten, in welchem er endet, vom gleichen Typ sein. Dies stellt eine zusätzliche Regel dar, welche dem Compiler-Programm eigen ist. Da die Verbindungen zwischen Knoten Datenströme repräsentieren anstelle einer physikali-

schen Verdrahtung, ist es zulässig, dass die Verbindungen zwischen Icons sich mit anderen Verbindungen oder selbst mit anderen Icons ohne eine nachteilige Wirkung kreuzen. Das rührt daher, dass der Datenstrom einen Ausgabeknoten und einen Eingabeknoten aufweist und nicht von irgendeiner dazwischen liegenden Stelle beeinflusst wird. Die Koordinaten der Quellen- und Zielknoten auf einem beliebigen Koordinatensystem werden verwendet, um Software-"Zeiger" zu erzeugen zu Quell- und Zieldatenpuffern zur Ausführung der Signalverarbeitungsfunktionen auf den Daten, auf die "gezeigt" wird.

- 10 Ebenso beinhaltet von einem Icon eine Icon-Spezifikation, bei der der Icon selbst unzureichend ist, um die gesamte Funktion des Icons zu beschreiben. Zum Beispiel hat ein Verstärker eine Funktion, $\text{Ausgang} = G \times \text{Eingang}$, wobei G die Verstärkungsleistung des Verstärkers darstellt. Um jedoch die Verstärkungsleistung, die vom Benutzer angegeben ist, zu aktivieren, kann die Verstärkungsleistung eingegeben werden als vorbestimmter Parameter mittels der Icon-Spezifikation.

- Es wird gesehen werden, dass unter Verwendung der obigen Vorgehensweise eine Bibliothek von vorbereiteten Icons vorbereitet werden kann, mit ihrem eigenen Programm, um die mathematische/signalverarbeitende Funktion des Icons auszuführen. Außerdem kann dieselbe Vorgehensweise wieder angewandt werden, wenn es wünschenswert ist, ein Programm zu kombinieren, das mathematische/Signalverarbeitungsfunktionen ausführen soll auf einem Feld, das von miteinander verbundenen Icons gebildet wird. Um ein einfaches Beispiel von einem Feld anzunehmen, sei ein Feld betrachtet, welches zwei Eingaben A und B und eine Ausgabe C aufweist, wobei das Feld aus einem Addierer besteht, der Eingänge A und B aufweist und der Ausgang des Addierers verbunden ist mit einem Verstärker, der eine Verstärkungsleistung G aufweist, wobei der Ausgang des Verstärkers den Ausgang des Feldes darstellt. Die mathematische/signalverarbeitende Funktion ist $C = G \times (A + B)$. Der graphische Compiler der bevorzugten Ausführungsform erzeugt Maschinencodes auf dieselbe Weise.

wie es ein FORTRAN-Compiler tun würde, wenn er im Wesentlichen dasselbe Statement erhält in Source-Code, geschrieben in der FORTRAN-Sprache.

Anhang I stellt ein Programmfragment dar von Code, der es dem Benutzer erlaubt zu entscheiden, was die Icon-Spezifikation sein wird für ein gegebenes Icon. Der „Pop-up“-Fensterabschnitt der Anzeige, in welchem der Benutzer den numerischen Wert/die numerischen Werte, die spezifiziert werden sollen, eingibt, wird ein „Blatt“ genannt.

- 10: Ebenso stellt das Programmfragment von Anhang II ein Beispiel dar von einem Programm, welches verwendet wird zum Kompilieren eines Icons. Der Code, der beim Kompilieren eines Feldes verwendet wird, ist ähnlich.

- 15: Anhang III ist ein Codefragment, welches ein Beispiel darstellt von einem digitalen Signalverarbeitungs-(DSP)-Implementationscode. Dieser Code erlaubt, dass digitale Signale, welche die Ausgabe von tatsächlichen elektronischen Vorrichtungen replizieren (wie dargestellt durch ein Icon oder ein Feld), erzeugt werden.

- 20: Schließlich stellt Anhang IV ein Codefragment dar von der Zeitgabe oder der Sequenz, die in der digitalen Signalverarbeitung verwendet werden. Dies stellt die zeitgerechte Vollendung der Berechnungen sicher in der Zeit zwischen aufeinanderfolgenden Abtastungen.

- 25: Der Betrieb in „Echtzeit“ wird unterstützt durch die Art, in welcher analoge Eingangssignale erfasst werden. Der zulässige ± 10 V Eingabebereich wird dargestellt durch eine 12-Bit-Zahl und das analoge Eingangssignal wird abgetastet mit einer Abtastfrequenz von 1 bis 60.000 Punkten oder Abtastungen pro Sekunde. Die genaue Abtastfrequenz wird angegeben oder ist auswählbar vom Benutzer.

Als Folge dieses Abtastschemas müssen alle Programmierschritte, die erforderlich sind, um eine bestimmte Funktion eines Icons oder eines Feldes zu emulieren, abgeschlossen sein und damit eine Ausgabe erzeugen in der Zeit, die verfügbar ist zwischen Abtastpunkten. Auf diese Weise wird die Ausgabe für einen gegebenen Abtastpunkt berechnet und damit erzeugt vor der Annahme der Information für den nächsten Abtastpunkt. Diese Prozedur erlaubt einen Echtzeitbetrieb unter der Voraussetzung, dass wenn die Berechnungszeit die Abtastperiode überschreitet, entweder zusätzliche Computer-Ressourcen in der Form eines zusätzlichen Prozessors und/oder Speichers bereit gestellt werden müssen, um die Berechnungszeit zu vermindern oder die Abtastfrequenz verringert werden muss, womit die Zeit für eine Berechnung erhöht wird. Die zunehmende Prozessleistung moderner Computer bedeutet, dass in der Praxis jegliche Beschränkung der Abtastfrequenz von keiner praktischer Bedeutung ist.

In Verbindung mit dem oben Gesagten wird es ebenso gewürdigt werden, dass die Berechnung, die von dem Computer ausgeführt werden soll, Rückkopplung einer Ausgabe beinhalten kann zu einer Abtastzeit, die dann die Eingabe für eine nachfolgende Berechnung zur nächsten Abtastzeit darstellt. Die nachfolgende Berechnung muss jedoch innerhalb der Abtastperiode abgeschlossen werden.

Fig. 8 veranschaulicht zwei verhältnismäßig einfache Felder. Die Eingabe- und Ausgabewellenformen von und zu diesen Feldern sind jeweils in Fig. 19 dargestellt. Man wird sehen, dass die Eingabe und die Ausgabe des ersten Feldes eine analoge Spannung umfassen von einem ECG. Für das andere Feld umfassen die vier Ausgaben die Ausgabe des spannungsgesteuerten Oszillators und die Ausgabe, wenn sie jeweils durch einen Tiefpassfilter, einen Hochpassfilter und einen Bandpassfilter geführt wird. Es sei bemerkt, dass, da diese Felder in keiner Weise miteinander in Beziehung stehen, obwohl all die Wellenformen gleichzeitig erzeugt und/oder dargestellt werden können, es für die Ergebnisse nicht notwendig ist, in irgendeiner Weise verbunden zu sein.

Bezug nehmend nun auf Fig. 10 veranschaulicht diese Zeichnung das Feld, das zuerst gezeichnet und dann kompiliert wird, um einen linearen Interpolations- schaltkreis zu emulieren. Die konstante Spannungsquelle 45 wird angewandt als
 57 Eingang auf drei spannungsgesteuerte Oszillatoren 46-48, die jeweils einen Sinus- Wellenausgang, einen Quadrat-Wellenausgang und einen Rampenausgang auf- weisen. Die Ausgabe der Sinuswelle wird auf 3 Hz eingestellt, während die Aus- gabe der anderen zwei Generatoren 47 und 48 eingestellt wird auf die Abtastfre- quenz von 20 Hz. Die Ausgabe des Sinus-Wellenoszillators 46 wird verwendet,
 10 um jede der drei Anzeigen 50-52 anzusteuern. Die Ausgabe des Quadrat- Wellenoszillators 47 wird verwendet, um zwei Abtast- und Halteschaltkreise 53, 54 anzusteuern, die getrennt sind von einem Zeitverzögerungsschaltkreis 55, der eine Verzögerung aufweist, die im Wesentlichen gleich ist der Periode zwischen dem Abtasten. Dies stellt sicher, dass die Ausgaben der zwei Abtast- und Halte- schaltkreise 53, 54 die Ergebnisse aufeinander folgender Abtastungen darstellen.

Eine weitere Spannungsreferenz 56 wird eingestellt auf ein Volt und umfasst eine Eingabe an einen Verminderer 57. Die lineare Interpolation wird ausgeführt durch die zwei Multiplizierer 58, 59 und den Addierer 60. Die Multiplizierer multipli-
 20 zieren den Rampen-Gradienten mit dem richtigen Verhältnis, das bestimmt wird durch die Amplitudendifferenzen von aufeinander folgenden Abtastungen, welche in den Schaltkreisen 53, 54 gehalten werden.

Die „Eingabe“ Sinuswelle bei 3 Hz, welche die Bildschirmanzeige 50 darstellt, ist
 25 in Fig. 11 veranschaulicht, ebenso wie die abgetastete Sinuswelle, die die Anzeige 51 darstellt. Die lineare Interpolation, welche erzeugt wird von der abgetasteten Sinuswelle stellt die Anzeige 52 dar und ist ebenso veranschaulicht in Fig. 11.

Fig. 12 veranschaulicht ein Feld, welches eine phasenverriegelte Schleife bildet. Eine Spannungsreferenz 61 stellt wiederum die Eingabe eines spannungsgesteuerten Sinuswellenoszillators 62 dar, wobei die Referenzspannung von 61 so gesteuert ist, dass die Frequenz, die vom Oszillator 62 erzeugt wird, eingestellt ist auf 51 Hz. Die Ausgabe des Oszillators 62 wird benutzt, um einen Pulsschaltkreis 63 anzusteuern. Die Ausgabe des Pulsschaltkreises 63 steuert ihrerseits drei Anzeigen 64-66 und bildet das Anzeigesignal für die Anzeige 64.

Zusätzlich wird die Ausgabe des Pulsschaltkreises 63 verwendet, um einen Abtast- und Halteschaltkreis 67 anzusteuern, der einen Eingang aufweist, der gebildet wird vom Ausgang des spannungsgesteuerten Sinuswellenoszillators 68, der so eingestellt ist, dass er eine Mittelfrequenz von 50 Hz aufweist. Die Ausgabe des Spannungssteuerungszosillators 68 bildet die Ausgabe des Abtast- und Halteschaltkreises 67, der im Wesentlichen die Phasendifferenz darstellt zwischen den Signalen von den Oszillatoren 68 und dem Pulsschaltkreis 63. Dieses Fehlersignal wird angezeigt durch die Anzeige 66, wobei die Ausgabe des Spannungssteuerungszosillators 68 angezeigt wird von der Anzeige 65. Die drei Anzeigen 64, 65 und 66 von Fig. 12 sind jeweils in Fig. 13 erläutert.

Ein Feld zur Lösung einer Differentialgleichung zweiter Ordnung ist veranschaulicht in Fig. 14. Hier werden drei Rückkopplungsschleifen FB1 - FB3 bereit gestellt. Um die Anfangsbedingungen einzustellen, wird eine Spannungsreferenz 70, die auf ein Volt eingestellt ist, verwendet, um eine Eingabe bereit zu stellen an einen Abtast- und Halteschaltkreis 71 und die invertierende Eingabe eines Komparators 72. Die Ausgabe des Abtast- und Halteschaltkreises 71 wird zurückgeführt zum Komparator 72 und ebenso zu einem der beiden Multiplizierer 73, 74. Die Ausgabe der Multiplizierer 73, 74 wird zusammengezählt im Addierer 75 und multipliziert mit der Verstärkungsleistung des Verstärkers 76, bevor sie integriert wird durch den ersten von zwei Integratoren 77, 78. Ein weiterer Verstärker 79 und Multiplizierer 80 vervollständigen den Schaltkreis. Die Ausgaben jedes der

Integratoren 77 und 78 bilden die Wellenformen, die von den beiden Anzeigen 81 und 82 angezeigt werden, welche beide angesteuert werden von der Ausgabe des ersten Integrators 77.

Die Ergebnisse sind veranschaulicht in Fig. 15 und zeigen, dass sowohl die Anfangslösung, die den Anfangsbedingungen entspricht, als auch die Ausgabe, welche die Lösung der Differentialgleichung darstellt, endlos läuft ohne ersichtlicher Verluste oder Gewinne innerhalb der Begrenzung von Quantisierungsfehlern. In einer Realisierung durch einen tatsächlichen elektronischen Schaltkreis, welche die Wirkung der Lösung der Differentialgleichung zweiter Ordnung hervorbringen soll, würde die Verwendung tatsächlicher Kondensatoren in Verlust oder fortschreitenden Abfall oder Anstieg des Ausgabesignals münden. Mit der oben beschriebenen Anordnung gibt es jedoch, weil die Lösung fortwährend berechnet wird, weder einen ersichtlichen Abfall noch einen instabilen Anstieg, der zu einer Sättigung führt. Dies stellt eine wesentliche Verbesserung gegenüber Analogcomputern nach dem Stand der Technik dar.

Fig. 16 ist ein Beispiel eines komplexeren Feldes, das kompiliert werden kann in Übereinstimmung mit der bevorzugten Ausführungsform der vorliegenden Erfindung. Es wird aus Fig. 16 ersichtlich werden, dass vier schnelle Fourier-Transformationen, sechs Differenzialsignalgeneratoren, numerische Anzeigen, Balkendiagramme, vier Referenzspannungen, vier 40 dB/Dekadenfilter, ein "Fläche unter der Kurve"-Berechner und verschiedene Komparaoren, Multiplizierer und Ähnliches alle bereitgestellt werden.

Weil die Ergebnisse, wie z.B. die Wellenformen 11, erläutert in Fig. 6, im Speicher gespeichert werden, ist der Benutzer in der Lage, diese gespeicherten Echtzeitergebnisse wieder abzuspielen. Darüber hinaus können verschiedene Abschnitte dieser Ergebnisse extrahiert werden für irgendeinen besonderen Zweck.

und selbst in die Vorrichtung zurückgeführt werden unter Verwendung eines "Von-Platte"-Icons. Dies erlaubt, dass die Signalverarbeitung so angepasst wird, dass ein gewünschtes Merkmal des Eingangssignals detektiert wird. Das gespeicherte Ergebnis kann ebenso fortwährend in Echtzeit wiedergegeben werden oder schneller oder langsamer, wie gewünscht. Ebenso können Anfangsbedingungen aufgestellt werden unter Verwendung des vorher gespeicherten Ergebnisses von einem ersten Feld als die Anfangseingabebedingung für ein zweites Feld.

Weiterhin können, weil die Ergebnisse gespeichert sind, die gespeicherten Daten editiert und exportiert werden auf Spreadsheets, graphische oder statistische Dienstprogramme, wie z.B. EXCEL, LOTUS 123 und Ähnliches. Dies erlaubt es, die graphischen Ergebnisse von Experimenten zu kombinieren mit Text, der die Natur der Experimente und die Natur der Schlussfolgerungen beschreibt.

Die Fenster, wie diejenigen, die in Fig. 8 angegeben sind, sind z.B. ebenso kompatibel mit Multimedia-Standards für den IBM PC. Die Spezifikation für den Fenster-Icon kann eingestellt werden, "PAL" zu sein, welches ein herkömmliches Signal von einer Videokamera empfängt, konvertiert und dann das Videosignal auf dem Bildschirm anzeigt. Falls es gewünscht wird, kann dieses Videosignal geframegrapped werden zu Videodaten, die gespeichert werden können im Computerspeicher. Falls gewünscht, kann das herkömmliche Videokamerasignal auch gleichzeitig auf einem VCR gespeichert werden. Die gespeicherten Videodaten können wieder abgespielt werden auf dieselbe Weise wie jede andere Wellenform, die von der Vorrichtung aufgenommen wurde. Auf diese Weise kann eine Videobildkorrelation mit anderen Signalwellenformen erreicht werden. Die Zeitmarke, welche auf das herkömmliche VCR-Videoband aufgebracht ist, wird in dieser Ausführungsform synchronisiert mit den Signaldaten, die von der Vorrichtung aufgenommen und darin gespeichert werden. Diese Anordnung ist besonders vorteilhaft, da sie erlaubt, elektrische Ergebnisse klar zu identifizieren mit der optischen Aufzeichnung der Ereignisse, welche die Ergebnisse erzeugt haben.

Die Bereitstellung von Icon-Spezifikationen bedeutet, dass die Vorrichtung besonders anpassungsfähig ist. Zum Beispiel kann ein Feld gebildet werden mit einer Ansicht zur Ausführung einer Analyse von z.B. den elektrischen Spannungen, die vom menschlichen Herzen erzeugt werden. Falls als Ergebnis der Analyse der Ergebnisse von diesen Experimenten gedacht ist, dass ein bestimmtes Arzneimittel eine bestimmte vorteilhafte Wirkung erzeugt, dann muss das Feld selbst nicht geändert werden, um es zu erlauben, dass dieselbe Messung ausgeführt werden kann mit einer Ratte, welcher das Arzneimittel verabreicht wurde. Alles, was erforderlich ist, geändert zu werden, ist die Spezifikation derjenigen Icons, die verwendet werden z.B. als eine Referenz. Zum Beispiel können das tatsächliche Spannungsniveau und die Frequenz angepasst werden, um den verschiedenen elektrischen Ausgängen Rechnung zu tragen (z.B. Spannungsniveaus und verschiedene Pulsraten) zwischen Menschen und Ratten.

15

Weiterhin bedeutet die Fähigkeit, Felder beliebig zu erzeugen, dass die Vorrichtung in der Lage ist, die Beziehung oder Unabhängigkeit zwischen Signalen zu messen und auf diese Weise eine weitere Dimension zu den Ergebnissen hinzuzufügen, welche erzielt werden können. Zum Beispiel kann eine Herzrate korreliert werden mit oder multipliziert werden mit einer Beatmungsrate eines Patienten, um ein drittes Signal zu ergeben, das betrachtet werden soll als Signatursignal, welches ein Ergebnis darstellt, welches dem Benutzer wichtig ist. Viele solche Signale, selbst von gemischten Einheiten, können auf diese Weise assoziiert werden, falls nötig geeignet skaliert werden und dann verglichen werden mit vorher im Speicher gespeicherten Daten.

20

Dem Fachmann wird es offenkundig erscheinen, dass in dem oben beschriebenen System alle Funktionen integriert sind, die erforderlich sind für eine analoge Workstation. Bis zum heutigen Tag gab es noch kein vollständig integriertes Sys-

ferm. Stattdessen gab es eine Vielfalt von partiellen Funktionen. Insbesondere stellen die folgenden Merkmale Komponententeile des integrierten Ganzen des wissenschaftlichen Emulators der bevorzugten Ausführungsform dar:

5 1 Ein Mehrzweck-/Mehrfunktions-Analogmodul, das eingibt/ausgibt digitale und analoge Signale und andere Funktionen.

2 Ein Echtzeit-Video, gleichzeitig sowohl in PAL oder Bildform- und Wellenformanzeigen, in bis zu 40 Fenstern auf einem Bildschirm sind verfügbar. Die
10 oszillographischen Anzeigen (eine Art von Anzeige-Verfahrensweise) können Abstraten in der Größenordnung von 25 KHz oder besser anzeigen.

3 Die Signalverarbeitung wird nicht gesteuert durch eine sequentielle syntaktische "Leitungssprache" im Prosastil (von Neumann), sondern durch einen
15 einfach verstandenen und sofort verwendeten parallelen graphischen Compiler, der von Nicht-Programmierern verwendet werden kann.

4 Wellenformen und graphische Felder in Bildform werden sofort zu bestehenden PC- Werkzeugen transportiert, wie z.B. Wortprozessoren, Spreadsheets
20 und "Offline"-Analysesoftware für die Berichterzeugung/Aufzeichnungen, etc.

5 Auf nicht-flüchtigen Speicher kann derart zugegriffen werden, dass er als "Quelle" von Daten dient, die zurückgeführt werden sollen in ein Feld für andere Ergebnisse. Ebenso ist das Korrelieren eines Wellenformmusters, das auf der
25 Platte gespeichert ist, mit einer beliebigen ankommenden Wellenform möglich, um eine Gestaltdetektion zu erreichen. (Template matching).

6. Netzwerkcompatibilität. Das System erlaubt es, dass seine verschiedenen Funktionen über ein Netzwerk verteilt sind. Das heißt, eine Speicherung kann ausgeführt werden in einem PC in einem Büro, oder Wellenformdaten können gespeichert werden in einem PC, um auf einem anderen angezeigt zu werden.
5. Ebenso können Programme und Daten über das Netzwerk transportiert werden zu einem anderen ähnlichen wissenschaftlichen Instrumentenemulator zur Analyse. Die Daten und Programme sind verbunden oder "gebündelt" zur Netzwerkübertragung.
10. In der bevorzugten Ausführungsform nimmt das System die Form einer Signalverarbeitungsvorrichtung an, die eine Vielzahl von proprietärer Hardware und Software umfasst, die in einem IBM PC enthalten ist mit Datenerfassungsverstärkern, welche in Floppy-Disk-Schächten untergebracht sind und Prozessor-PCBs und Echtzeit Videokarten, die aufgesteckt sind auf dem PC-Systembus. Die Vor-
15. richtung in einer Konfiguration enthält zusätzlich 2 Mehrzweckprozessoren und weiter 2 Signalprozessoren. Diese kombinierten Prozessoren werden programmiert durch die Verwendung eines graphischen Compilers auf dem VGA-Bildschirm auf eine solche Weise, dass der PC-Host (betrieben bei etwa 3.000.000 Instruktionen pro Sekunde) das residente Vorrichtungssubsystem (betrieben bei
20. etwa fünfzehn Millionen Instruktionen pro Sekunde) als eine Erweiterung seines Speichers ansieht. All die anderen Prozessoren sehen einander auch als Erweiterungen ihres eigenen Speichers. Auf diese Weise kann die Multitasking-Software mehrfache Prozessoren verwalten, wobei jeder Task auf den verschiedenen Pro-
25. zessoren mit dem anderen Task/den anderen Tasks kommuniziert und/oder mit Prozessor(en) durch Weitergeben von Zeigern in den gemeinsamen Speicher. Auf diese Weise wird keine besondere Kommunikationshardware oder software erforderlich. Der graphische Compiler, der ausführbaren Objektcode für das installierte Subsystem kompiliert, erlaubt den strengen Gebrauch von Mathematik, die so schnell verarbeitet wird, dass sie Echtzeit simuliert, wobei alle notwendigen
30. Verarbeitungsschritte innerhalb einer einzigen Abtastzeit durchgeführt werden.

Zusätzlich zu der funktionalen Zulänglichkeit des Programmierungssystems des graphischen Compilers ist das Programmsystem, das Icons benutzt, auf einer solchen Weise aufgebaut, dass kompilierte Icons selbst zusammengesetzt sein können aus einfachen Icons. Dies erlaubt, dass ein mathematischer Aufbau angewandt werden kann auf zwei Arten von Datenströmen, die bei der Verarbeitung vorkommen. Einer dieser Typen von Datenströmen ist der "erfasste Datenstrom" von dem Analog-zu-Digital-Umwandlungsprozess und der andere Typ ist ein "Ansteuerungsstrom". Ein Ansteuerungsstrom ist ein Signal, das erfasst wird von dieser Zwischenumwandlung und mit dem andere Signale, von denen gewünscht wird, dass sie angezeigt werden in einer Zeitbeziehung, oder es wird erzeugt von einer Zeitbasis, die in dem Feld aufgestellt ist. Am häufigsten wird der Ansteuerungsstrom erzeugt durch eine Verarbeitungsfunktion, wobei die Verarbeitung auf einem oder mehreren Kanälen ein Signal erzeugt, das Anzeigen ansteuern soll, eine Verarbeitung anzusteuern, wie z.B. eine Mittelwertbildung oder eine Speicherung anzusteuern. Im Allgemeinen ist es die komplexe Ansteuerungsfähigkeit des Systems, die vereinfacht durch den graphischen Compiler dargestellt ist, welcher es erlaubt, dass Parameter akkumuliert werden, welche verschiedene Wellenformen oder Epochen von Wellenformen charakterisieren. Dies erlaubt deren Auswahl durch Experten, um einen "Beispielsatz" zu begründen.

INDUSTRIELLE ANWENDUNG

Die Kostengünstigkeit der Vorrichtung der bevorzugten Ausführungsform kann aus dem Folgenden in Betracht gezogen werden. Die Vorrichtung kann als irgendeine der folgenden Vorrichtungen oder als irgendeine Kombination der folgenden Vorrichtungen (oder Vielfachen davon), die zusammen in einem Feld angeordnet sind, arbeiten.

	VORRICHTUNG	KOSTEN A\$
1.	4-Kanaldigramm-Recorder (500 Hz/Kanal)	10.000
2.	FFT-Analysierer (1 Hz-20 KHz, max. 512 Punkte, 200 Analysen/sec.)	13.000
3.	Frequenzzähler (0,01% Genauigkeit, 0,01 Hz-10 MHz)	1.000
5.	4. Funktionsgenerator (Sinus-, Quadrat-, Rampen- und Dreieck-Wellenformen, 0,01 Hz - 2 MHz)	5.000
5.	Datenaufzeichnung (PC-basiert)	4.000
6.	Leistungsüberwacher (Volt x Spannung, isolierte Eingänge)	4.000
7.	2-Kanalübergangsrecorder	3.500
10.	8. Elektrophysiologischer Überwacher (ECG, EEG, EMG, ERP)	20.000

Die obigen Ausrüstungsteile ergeben Kosten von etwa 60.500 A\$, wohngegen den Wiederverkaufspreis eines wissenschaftlichen Instrumentenemulators, gemäß der bevorzugten Ausführungsform, welcher in der Lage ist, die obigen Funktionen zu emulieren, in der Nähe von 15.000 A\$ liegt (unter der Annahme, dass der Benutzer einen geeigneten Personal Computer besitzt, der sowohl die notwendige Hardware als auch die notwendige Software ablaufen lassen kann).

Weiterhin ist die große Anzahl paralleler Eingänge in dem integrierten System gut geeignet für das Studium und die Analyse paralleler Systeme, wie z.B. von Ökosystemen, Biosystemen, Maschinensystemen, etc.

Das Vorangegangene beschreibt lediglich eine Ausführungsform der vorliegenden Erfindung und Modifikationen, die dem Fachmann offensichtlich sind, können daran vorgenommen werden, ohne vom Schutzbereich der vorliegenden Erfindung abzuweichen.

000000

-29-

COPYRIGHT-HINWEIS

Die Programmlisten, welche in den Anhängen I-IV enthalten sind, unterliegen dem Copyright, welches vom Anmelder gehalten wird, und sie dürfen nicht reproduziert werden in jeglicher Form ohne die ausdrückliche vorherige schriftliche Erlaubnis des Anmelders.

03-03-12

- 30 -

ANHANG I

Code für Seiten.

Der folgende Code wird verwendet, um eine Seite zu bestimmen, in welcher der Benutzer die Parameter für einen gesteuerten Pulsgenerator bestimmt.

```

/*****
 * AMLAB - Associative Measurement Laboratory
 *
 * PULSE Icon Overlay Sheet
 *
 */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <dir.h>
#include "library.h"
#include "pulsegen.h"

/*****
 * Window Dimension Constants
 *****/
#define WINDOW_WIDTH 55
#define WINDOW_HEIGHT 21
#define USER_X_SIZE WINDOW_WIDTH
#define USER_Y_SIZE (WINDOW_HEIGHT - 1)

#####
### PULSE Sheet Parameter Definition
#####

typedef struct
{
    ICON_PARAM ip;
    PULSE_ICON_PARAM pi;
} PULSE_SHEET;

PULSE_SHEET pulse_vars =
{
    {ICON_PARAMETER_REC,
    sizeof(PULSE_SHEET), {0}, sizeof(PULSE_ICON_PARAM)},
    {1000, MAX_STATE_LEVEL, MIN_STATE_LEVEL, 0,
    POSITIVE_EDGE, 0, 0, NORMAL_TRIGGER}
};

```

05.03.02

```
#####
**** PULSE Sheet WINDOW Definition
#####
/*
.....
PULSE sheet Variables
.....
*/
/* Local string Arrays */
char title[] = "PULSE GENERATOR SHEET";

/* ##### Sheet Title Definition ##### */
TITLE pulse_sheet_title =
{
    NULL,
    -1,
    BLACK, WHITE,
    title,
    NULL,
};

/* ##### Sheet Control Definition ##### */
CONTROL pulse_sheet_control =
{
    NULL,
    -1,
    IWHITE, GREY,
    LOGO,
    0.0,
    0, NULL,
    0.0, 0.0,
    0.0, 0.0,
};

/* ##### Edit Region Structure Definitions: ##### */
/*
int check_duration(int);
EDIT_DEFN pulse_duration =
{
    UNSIGNED_INTEGER,
    5,
    BLACK, WHITE, IWHITE, GREY,
    0.0,
};
```


000000

```
"Pulse ON State Duration (Samples):"  
&pulse_vars.pa.duration;  
check_duration;  
NULL;  
};
```

```
int check_on_level(int);  
EDIT_DEFN pulse_on_level =  
{  
  INTEGER,  
  6,  
  BLACK,WHITE,WHITE,GREY,  
  0,0,  
  "Pulse State Levels (mV) - ON:",  
  &pulse_vars.pa.on_level,  
  check_on_level,  
  NULL,  
};
```

```
int check_off_level(int);  
EDIT_DEFN pulse_off_level =  
{  
  INTEGER,  
  6,  
  BLACK,WHITE,WHITE,GREY,  
  0,0,  
  "- OFF:",  
  &pulse_vars.pa.off_level,  
  check_off_level,  
  NULL,  
};
```

```
int check_trigger_level(int);  
EDIT_DEFN pulse_trigger_level =  
{  
  INTEGER,  
  6,  
  BLACK,WHITE,WHITE,GREY,  
  0,0,  
  "Trigger Level (mV):",  
  &pulse_vars.pa.trigger_level,  
  check_trigger_level,  
  NULL,  
};
```

000000

```
int check_edge(int);
EDIT_DEFN pulse_trigger_edge =
```

```
{
    CHAR,
    BLACK, WHITE, WHITE, GREY,
    0, 0,
    "Edge (+, -): ",
    &pulse_vars.pa.edge,
    check_edge,
    NULL,
};
```

```
int check_slew(int);
EDIT_DEFN pulse_trigger_slew =
```

```
{
    UNSIGNED_INTEGER,
    S,
    BLACK, WHITE, WHITE, GREY,
    0, 0,
    "Slew (mV/mS): ",
    &pulse_vars.pa.slew,
    check_slew,
    NULL,
};
```

```
int check_mode(int);
EDIT_DEFN pulse_trigger_mode =
```

```
{
    CHAR,
    MAX_TRIG_MODE,
    BLACK, WHITE, WHITE, GREY,
    0, 0,
    "Mode (SINGLE SWEEP, NORMAL): ",
    &pulse_vars.pa.trigger_mode,
    check_mode,
    NULL,
};
```

```
/* ##### Sheet Region Definitions ##### */
```

```
SCREEN_REGION pulse_sheet_regions[] =
```

```
{
    EDIT_CLASS,
```

08-02-02

= 34 =

4.2.0.0;
&pulse_duration;

EDIT_CLASS;
4.5.0.0;
&pulse_on_level;

EDIT_CLASS;
28.7.0.0;
pulse_off_level;

EDIT_CLASS;
4.10.0.0;
&pulse_trigger_level;

EDIT_CLASS;
12.12.0.0;
&pulse_trigger_edge;

EDIT_CLASS;
12.14.0.0;
&pulse_trigger_slew;

EDIT_CLASS;
4.17.0.0;
&pulse_trigger_mode;
};

#define NR_REGIONS (sizeof(pulse_sheet_regions)/
sizeof(SCREEN_REGION))

/* ##### PULSE Sheet Window Definition ##### */

AMWINDOW pulse_sheet_window] =
{
0.0.0.0,
(VIRTUAL_MAX_X/2),(VIRTUAL_MAX_Y/2),
WINDOW_WIDTH,WINDOW_HEIGHT,
0.0,
BLACK_WHITE,
NULL,
NULL,
NULL,
&pulse_sheet_title,

00:03:02

-35-

```
&pulse_sheet_control;  
0:NULL;  
NR_REGIONS pulse_sheet_regions  
};
```

```
/* ##### Error Messages ##### */  
ERROR_MSG pulse_sheet_errors[] = {  
    0 /*  
*/
```

```
    RANGE_ERROR,  
    "Pulse Duration Error\n\nThe Pulse\  
Duration specified is Invalid\n\nValid range is from 1 to 65535 samples",
```

```
    /* 1 */  
    RANGE_ERROR,  
    "ON State Level Error\n\nThe ON state\  
level specified for the pulse is Invalid\n\nValid Range is -10000 to 10000 mV",
```

```
    /* 2 */  
    RANGE_ERROR,  
    "OFF State Level Error\n\nThe OFF state\  
level specified for the pulse is Invalid\n\nValid Range is -10000 to 10000 mV",
```

```
    /* 3 */  
    RANGE_ERROR,  
    "Trigger Level Error\n\nThe Trigger\  
level specified is Invalid\n\nValid Range is -10000 to 10000 mV",
```

```
    /* 4 */  
    RANGE_ERROR,  
    "Edge Error\n\nThe Trigger Edge\  
specified is Invalid\n\nValid Edges are + or -",
```

```
    /* 5 */  
    RANGE_ERROR,  
    "Trigger Mode Error\n\nThe Mode\  
specified is Invalid\n\nValid Modes are SINGLE SWEEP or NORMAL",
```

```
    /* 6 */  
    RANGE_ERROR,  
    "Slew Rate Error\n\nThe Slew\  
rate specified is Invalid\n\nValid Range is 0 to 10000 mV/ms",
```

08-03-12

-36-

```

/*
=====
**** PULSE Sheet Definition
=====
*/

```

```

SHEET
    pulse_sheet =
    {
        pulse_sheet_window,
        &pulse_vars.p,
        pulse_sheet_errors
    };

```

```

/* Local Save Area of Global Variable Pointer */
GLOBAL_VARS *global_variables;

```

```

/*=====
PULSE Sheet Handler
=====*/

```

```

/*
SHEET *pulse_sheet_handler(GLOBAL_VARS *global)
{
    /* preset the global variables static */
    global_variables = global;

    return(sheet_handler(global, USER_X_SIZE, USER_Y_SIZE,
        NR_REGIONS, &pulse_sheet, pulse_sheet_regions));
}

```

```

/*=====
Check Duration (0)
=====*/

```

```

/*
int check_duration(int checkpoint)
{
    return(check_range(checkpoint, &pulse_vars.p, duration,
        MAX_UNSIGNED, 1, 0));
}

```

etc...

ANHANG II

Compiler-Elemente

Der folgende Code stellt ein Beispiel dar des Compiler-Elements, welches für einen gesteuerten Pulsgenerator verwendet wird.

```

/*=====
 *                               Pulsegen Icon Compiler Overlay
 *=====
 */
#include <stdio.h>
#include <stdlib.h>
#include "compiler.h"
#include "library.h"
#include "comms86.h"
#include "pulsegen.h"
/*=====
 *                               Local Definitions
 *=====
 */
void init_trigger_params(PULSE_ICON_PARAM *);

/*=====
 *                               Pulse Process
 *=====
 */
#define NR_PULSE_INPUTS 1
#define NR_PULSE_OUTPUTS 1

/* I/O Buffer Lists */
unsigned int pulse_input_ids[NR_PULSE_INPUTS] = {1};
unsigned int pulse_output_ids[NR_PULSE_OUTPUTS] = {0};

/* Cell Parameters. */
struct PULSE_struct pulse_param;

/* Processes in this Icon. */
struct proc_type pulse_proc =
{
    NULL, /* the next process defined */
    PULSE, /* the cell required */
    pulse_input_ids, /* array of input buffer numbers */
    pulse_output_ids, /* array of output buffer numbers */
    &pulse_param /* array of the initial values of the cells parameters */
}

```

00-00-00

= 38 =

```
0 /* the base xdrum address of variables for this cell */
1
/*
#####
### Trigger Process
#####
*/
#define NR_TRIGGER_INPUTS 1
#define NR_TRIGGER_OUTPUTS 1

/* Trigger input ID specifications */
unsigned int trig_input_ids[NR_TRIGGER_INPUTS] = {0};

/* Trigger output ID specifications */
unsigned int trig_output_ids[NR_TRIGGER_OUTPUTS] = {1};

/* Trigger parameters */
struct THRESH_struct trig_params;

/* Process Descriptor */
struct proc_type trigger_proc =
{
    &pulse_proc, /* the next process defined */
    THRESH, /* the cell id required */
    trig_input_ids, /* array of input buffer numbers */
    trig_output_ids, /* array of output buffer numbers */
    &trig_params, /* array of the initial values of the cells
parameters */
    0 /* the base xdrum address of variables for this
cell */
};

/*
#####
### PULSEGEN CONFIGURATION
#####
*/
unsigned int int_interconnect_ids[] =
{
    &trig_output_ids[0],
    &pulse_input_ids[0]
};
```

08-03-02

- 37 -

```
#define NR_INTERCONNECT_IDS (sizeof(int_interconnect_ids)/
                             sizeof(unsigned int*))
```

```
unsigned int *ext_input_ids[] =
{
    &trig_input_ids[0]
};
```

```
#define NR_EXT_INPUT_IDS (sizeof(ext_input_ids)/
                          sizeof(unsigned int*))
```

```
unsigned int *ext_output_ids[] =
{
    &pulse_output_ids[0]
};
```

```
#define NR_EXT_OUTPUT_IDS (sizeof(ext_output_ids)/
                           sizeof(unsigned int*))
```

```
/* PULSEGEN Dsp System Configuration */
/* ICON_CONFIG pulse_config =
(
    2, /* nr_processes */
    &trigger_proc, /* process list */
    0, /* nr_raw */
    NULL, /* raw list */
    0, /* nr_screen */
    NULL, /* screen list */
    0, /* nr_virtual */
    NULL, /* virtual list */
    NR_INTERCONNECT_IDS, /* number of int interconnect ID's */
    int_interconnect_ids,
    NR_EXT_INPUT_IDS, /* number of external input ID's */
    ext_input_ids,
    NR_EXT_OUTPUT_IDS, /* number of external output ID's */
    ext_output_ids,
    NULL, /* Overlay Struct */
    NULL, /* Icon parameter key */
    NULL, /* next_proc */
    NULL, /* next_raw */
    NULL, /* next_scrn */
    NULL, /* next_virt */
    NULL, /* next_icon */
);
```

SUBSTITUTE SHEET

000000

```

/*=====
Request Block Size
=====*/
*/
unsigned int mem_block_size = 0;

/*=====
Pulsegen Descriptor Handler
=====*/
*/
void *pulsegen(GLOBAL_VARS *gv, void *mem_block, ICON_PARAM
*ip)
{
PULSE_ICON_PARAM *pip;

/* Check for memory block request */
if (mem_block == NULL)
return(&mem_block_size);

/* Index pulse icon parameters */
pip = (PULSE_ICON_PARAM *) &ip[1];

/* Set the pulse state levels */
pulse_param.WIDTHPU = pip->duration;
pulse_param.HIGHPU = calc_AD_val(pip->on_level);
pulse_param.LOW_PU = calc_AD_val(pip->off_level);

/* Set the Trigger Cell Variables */
init_trigger_params(pip);

/* Return Configuration Pointer */
return(&pulse_config);
}

/*=====
Init Trigger Params
=====*/
*/
void init_trigger_params(PULSE_ICON_PARAM *pip)
{
/* Transfer the level parameter */
trig_params.THRETH = calc_AD_val(pip->trigger_level);

/* Preset trigger width */
}

```

03.03.02

trig_params.WINSTH = 1;

/* Transfer the Slew Parameter */
trig_params.SLEWTH = pip->slew;

/* Preset Dead Period */
trig_params.DEADTH = 0;

/* Transfer the Trigger Edge Setting */
if (pip->edge[0] == POSITIVE_EDGE)
trig_params.POSTTH = FLAG_POS_EDGE;
else
trig_params.POSTTH = FLAG_NEG_EDGE;

/* Transfer the One Shot Status */
if (strcmp(pip->trigger_mode, SINGLE_SWEEP_TRIGGER) == 0)
trig_params.ONESTH = FLAG_ONE_SHOT;
else
trig_params.ONESTH = FLAG_CONTINUOUS;

ANHANG III

Zellen-Code

Der folgende Code wird verwendet, um den Pulsgenerator auf dem DSP-Chip zu implementieren. Er ist geschrieben in TMS320 Assembler-Code.

Cell PULSE

Synopsis

- This cell generates a pulse once every time a trigger occurs.
- It has one input which is a trigger stream.
- It has one output which is the pulse train produced.

The cell has the following functioning:

read in the trigger buffer

for each point in the buffer:

if waiting for the trigger to go high:

if trigger is high:

change state to 1.
reset pulse width counter.

else if trigger is low

change to state 0

if (count > 0):

count--
output = high

else output = low

finish the cell

IDT PULSE
COPY WHERELIB

SYSTEM VARIABLES

REF	RG1	WORKING REGISTERS
REF	RG2	
REF	RG3	
REF	ONE	THE NUMBER ONE
REF	BUFSIZ	THE SIZE OF BUFFERS
REF	I	A COMMONLY USED LOOP COUNTER
REF	FRESH	THE ALLOC BASE PTR
REF	ZERO	ZERO

SYSTEM CALLS

REF	ENDCEL	THE END OF THIS CELL
-----	--------	----------------------

LIBRARY ROUTINES

REF	READCH	READ IN A CHANNEL
-----	--------	-------------------

THIS CELL'S VARIABLES

REF	TRIGPU	INPUT THE CELL'S TRIGGER
		INPUT
REF	OUTSPU	OUTPUT THE CELL'S OUTPUT
REF	BUFFPU	TEMP ARRAY USED TO BUFFER
		POINTS
REF	STATPU	STATIC STATE OF TRIGGER
		DETECTION
REF	WIDTPU	PARAM WIDTH OF PULSE

00-00-00

44

REF HIGHPU
REF LOWSPU
REF COUNPU

PARAM HIGH PULSE LEVEL
PARAM LOW PULSE LEVEL
STATIC THE NUMBER OF POINTS
LEFT IN THE PULSE

CONSTANTS

COPY CASPDEF

PULSE

INPUTS
TRIGPU

OUTPUTS
OUTSPU

DEF PULSE

PULSE

read in the trigger buffer

LARK AR1:BUFEPU
LAC TRIGPU
SACL RG1
CALL READCH

for each point in the buffer:

for(i=0; i<buf_size; i++)

LAR AR0:BUFSIZ
LARK AR1:BUFEPU
LAC OUTSPU
ADD ONE,1
SACL RG1
SXRAM RG1

LINE UP WITH DATA

08-03-02

-45-

PULSFI BANZ PULSCI

B ENDCEL
PULSCI

if trigger is high:-

• READ IN THE TRIGGER

LARP ARI
LAC *+0:ARO
BZ ELSE1

if waiting for the trigger to go high:-

LAC STATPU
BNZ DOOUT

change state to 1.
reset pulse width counter.

LAC ONE
SACL STATPU
LAC WIDTPU
SACL COUNPU
B DOOUT

else change to state 0

ELSE1ZAC
SACL STATPU

if (count > 0):

DOOUT LAC COUNPU
BLEZ ELSE2

count--
output = high

SUB ONE
SACL COUNPU

46

00000000

WXDRAM HIGHPU
B PULSFI

else output = low

NOP
ELSE2 WXDRAM LOWSPU
B PULSFI

finish the cell

END

- 49 -

ANHANG IV

DSP-System-Code

Der folgende Code implementiert das System, welches von DSPs in einer AMLAB-Umgebung verwendet wird:

MODULE- SYSTEM

.....

SYNOPSIS

- DUE TO THE PERFORMANCE REQUIREMENTS OF THE TMS320 IT
- IS NECESSARY TO IMPLEMENT EACH PROCESSING ELEMENT AS
- A CELL IN A INTERPRETER TYPE SITUATION. AN EXPLANATION
- OF THIS INTERPRETER FOLLOWS. NO ATTEMPT TO EXPLAIN
- WHY THINGS ARE DONE THE WAY THEY ARE IS GIVEN HERE AS
- THESE CAN BE FOUND IN THE ACCOMPANYING
- SPECIFICATIONS.

DEFINITION OF TERMS

- **CELL** A PROCESSING ELEMENT. THE TOTAL PROCESSING TO BE DONE IS BROKEN DOWN INTO MODULES THAT ARE EASY TO IMPLEMENT.
- **EVENT** AN EVENT IS SOMETHING THAT HAPPENS. IT MAY BE INTERNAL TO THE SYSTEM SUCH AS A BEAT IS PROCESSED TO A CERTAIN LEVEL OR EXTERNAL SUCH AS AN INTERRUPT. THIS EXECUTIVE IS AN EVENT DRIVEN ONE. IN THAT IT IS EVENTS WHICH ULTIMATELY DETERMINE WHICH SEQUENCES ARE RUN AND IN WHAT ORDER.
- **SEQUENCE** A PRE-DETERMINED ORDER IN WHICH CELLS ARE RUN.
- **ACTIVE** THE ACTIVE CELL IS THE CELL CURRENTLY BEING EXECUTED OR ABOUT TO BE EXECUTED.
- **CCB** A CELL CONTROL BLOCK. THIS CONTAINS ALL THE

05-10-02

RELEVANT INFORMATION ABOUT A CELL NEEDED BY THE SYSTEM:

- **ENDSCEL** THE ROUTINE WHICH TIDIES UP AFTER A CELL RELINQUISHES CONTROL, FINDS THE NEXT CELL IN THE SEQUENCE, AND INSTALLS THAT CELL.

DESCRIPTION OF THE SYSTEM

- THE SYSTEM COULD IN BROAD TERMS BE DESCRIBED AS AN INTERPRETER. CELL 0 IS THE HIGHEST LEVEL OF PROCESSING IN THE SYSTEM. IT DETERMINES WHICH SEQUENCES WILL BE RUN BASED ON THE INFORMATION CONTAINED IN THE STATUS. SEQUENCES IN TURN ARE COMPOSED OF CELLS. CELL 0 RUNS A SEQUENCE. THE SYSTEM TAKES THAT SEQUENCE AND RUNS EACH CELL IN TURN. INDIVIDUAL CELLS CAN CHANGE THE STATUS TO INDICATE TO CELL 0 WHAT IS THE RESULTS OF ITS PROCESSING. FURTHER MORE CELLS ARE ABLE TO ABORT THE CURRENT SEQUENCE WHICH RETURNS CONTROL TO THE CELL 0. CELL 0 IS DIFFERENT TO THE OTHER CELLS IN THAT IT ALWAYS SAVES A RETURN ADDRESS BEFORE RELINQUISHING CONTROL. THIS IS BECAUSE CELL 0 IS NEVER COMPLETED, RELINQUISHING CONTROL FREQUENTLY, AND RESTARTING WHERE IT LEFT OFF THIS ALLOWS US TO IMPLEMENT A MACRO LEVEL PROGRAM IN CELL 0, COMPOSED OF SEQUENCES WHICH IN TURN ARE COMPOSED OF CELLS.

- SEQUENCES ARE REFERENCED BY THE SEQUENCE REFERENCE LIST, WHICH, FOR EASE OF PROGRAMMING EXISTS AT THE BOTTOM OF EXTERNAL DATA RAM. EACH MEMBER IN THIS LIST POINTS TO A SEQUENCE LIST. EACH SEQUENCE LIST MEMBER IS A POINTER TO CELL CONTROL BLOCK. EVERY SEQUENCE HAS A SEQUENCE LIST. EACH CONSECUTIVE CELL IN A SEQUENCE IS POINTED TO BY A CONSECUTIVE MEMBER IN THE SEQUENCE LIST.

- A CELL IS RESTORED BY LOADING INTERNAL DATA MEMORY WITH THE CELLS STATIC VARIABLES. THESE ARE POINTED TO BY A MEMBER OF THE CCB. THE CELLS START ADDRESS IS ACCESSED BY USING THE CELL ID. ANOTHER MEMBER OF THE CCB, AS AN OFFSET TO A LOOK UP TABLE STORED IN PROGRAM MEMORY CALLED THE CSASB. A CELL IS SAVED BY THE REVERSE PROCESS. THE FINAL ELEMENT OF A CCB IS A CONSTANT. A CONSTANT IS A STATIC VARIABLE THAT DOES

09-03-02

- 19 -

NOT CHANGE ITS VALUE AND HENCE DOES NOT NEED TO BE
STORED AWAY.

IDENT SYSTEM
COPY WHERE LIB

STATIC VARIABLES USED BY THE SYSTEM

REF	CCBSET	A POINTER TO THE CURRENTLY ACTIVE CCB
REF	CURSSQ	A POINTER TO THE CURRENTLY ACTIVE SEQUENCE
REF	ONE	
REF	TABLE	A POINTER TO THE VARIABLES TABLE IN IORAM
REF	RG1, RG2	
REF	CSASB	THE CELL ADDRESS BLOCK
DEF	ABORT	LET OUTSIDE TASKS KNOW ABOUT THE ROUTINES
DEF	ABORT1	
DEF	ENDCEL	
DEF	RUN	

CONSTANTS USED BY THE SYSTEM

REF	CELVAR	THE START OF A CELLS VARIABLES IN INTERNAL RAM
REF	C0CCB	THE ADDRESS OF CELLO CCB

THE SYSTEM ENTRY POINT THIS IS ONLY USED ON START

UP

00000000

```

*****
DEF SYSTEM
PSEG
SYSTEM B ABORT: SYSTEM ENTRY POINT
*****

```

 SAVE THE ENVIROMENT OF THE CURRENT CELL

```

SAVE R0DRAM RG2.CCBSET
R0DRAM TABLE

```

```

R0DRAM RG1

```

```

R0DRAM RG1

```

```

LAR AR0RG1

```

```

LAR AR1CELVAR

```

```

SXGRAM TABLE

```

```

LARP 0

```

```

BANZ SAVELP

```

```

B SAVRET

```

```

SAVELP

```

```

LARP 1

```

```

WXDRAM LAR0

```

```

BANZ SAVELP

```

```

SAVRET

```

```

RET

```

READ IN THE CELL ID
 READ IN THE CURRENT
 CELLS VARIABLE TABLE
 READ IN THE NUMBER OF
 INSTALLED VARIABLES
 READ IN THE NUMBER OF
 STORED VARIABLES

SET UP THE POINTER TO
 THE VARIABLES

ARP = 0
 IF THERE ARE ANY MORE
 VARIABLES SAVE THEM

WRITE OUT THE NEXT
 VARIABLE
 ARE THERE ANY MORE

 RESTORE THE ENVIROMENT OF THE CURRENT CELL

000000

= S =

RESTOR

R0DRAM RG2.CCBSPT
R0DRAM TABLE

READ IN THE CELL ID
READ IN THE CURRENT
CELLS VARIABLE TABLE
ADDR.
READ IN THE NUMBER OF
VARIABLES

R0DRAM RG1

LAR AR0, RG1

LAR AR1, CELVAR

SET UP THE POINTER TO
THE VARIABLES

SXRAM TABLE

LARP 0

BANZ RESTLP

ARP = 0

IF THERE ARE ANY MORE
VARIABLES SAVE THEM

B RESRET

RESTLP

LARP 1

R0DRAM *+, AR0

WRITE OUT THE NEXT
VARIABLE
ARE THERE ANY MORE?

BANZ RESTLP

RESRET

LACKB CSASB

FIND THE RETURN
ADDRESS FROM THE CSASB
THE CELLS ID
READ IN THE RETURN
ADDRESS

ADD RG2
TBLR RG1

LAC RG1
PUSH
RET

PUSH IT ONTO THE STACK
AND GO THERE

0-0-02

-2-

* ABORT THE CURRENTLY ACTIVE SEQUENCE AND RETURN TO
* CELL 0 THIS ROUTINE SHOULD BE BRANCHED TO

ABORT

CALL SAVE

SAVE THE ENVIROMENT OF THE
CURRENTLY ACTIVE CELL

ABORT1

LACB

CCCB

SET THE CCB POINTER TO CELL 0
CCB

SACL CCBSP

B

RESTOR

AND RESTORE CELL 0

* END THIS CELL AND INSTALL THE NEXT ONE IN THE
* SEQUENCE THIS ROUTINE SHOULD BE BRANCHED TO

ENDCEL

CALL SAVE

SAVE THE ENVIROMENT OF THE
LAST CELL

* FIND THE ADDRESS OF THE NEXT CELL IN THE SEQ

RDRAM CCBSET CURSSQ

LAC CURSSQ

INCREMENT THE SEQ POINTER

ADD ONE

SACL CURSSQ

B

RESTOR

INSTALL THE NEXT CELL

000000

THIS ROUTINE IS ONLY EVER CALLED BY CELL 0. IT
STARTS OFF A NEW SEQUENCE.

RUN RXDRAM CURSSQ, RGIRGI CONTAINS THE SEQUENCE
NUMBER WHICH IS A
DIRECT OFFSET TO THE
SEQUENCE POINTER.
GET THE RETURN ADDRESS
POP SACL RG2
LACKB CSASB AND WRITE IT OUT TO THE FIRST
LOCATION IN THE
CSASB
B ENDCEL AND PERFORM ENDCEL WHICH
WILL SAVE THE ENVIROMENT OF
CELL 0 AND THEN INSTALL THE FIRST CELL IN THE SEQUENCE

PEND

09.03.02

692 32 869 6 08
Associative Measurement PTY. LTD.

06. März 2003
S 17755 EP/DE AI/Sn/bb

Ansprüche

- 5 1. Wissenschaftlicher Instrumentenemulator, aufweisend:
einen Computer (1) mit einem Speicher (3), der mit einem Prozessor (2)
gekoppelt ist;
ein multifunktionales Eingangs-/Ausgangs-Signalmodul (6), welches zu-
mindest einen analogen Signaleingang (7) mit Digitalisierungsmitteln (37)
10 hierzu verbunden und zumindest einen Signalausgang (8) aufweist und
konfigurierbar ist mit Bezug auf zumindest eine Abtastfrequenz des analo-
gen Signaleingangs (7);
eine Videoanzeige (10), die mit einem Videoanzeigegenerator (50) gekop-
pelt ist, wobei der Videoanzeigegenerator (50) mit dem Prozessor (2) ge-
15 koppelt ist;
ein Bibliotheksprogramm, das in dem Speicher (3) gespeichert ist, wobei
das Bibliotheksprogramm eine Mehrzahl von Instrumentationseinheiten
aufweist, wobei jede geeignet ist, auf der Videoanzeige (10) angezeigt zu
werden, und jede eine vorbestimmte Signalverarbeitungsfunktion aufweist;
20 ein Set-Up-Programm, das in dem Speicher (3) gespeichert ist, und durch
den Prozessor (2) ausgeführt wird, um einem Bediener zu erlauben, aus-
gewählte Instrumentationseinheiten miteinander zu verbinden, um ein Feld
von miteinander verbundenen Instrumentationseinheiten zu bilden, wobei
eine Abtastfrequenz für das multifunktionale Eingangs-/Ausgangs-
25 Signalmodul (6) spezifiziert ist, wobei das Set-Up-Programm Signalkück-
führungsverbindungen in dem Feld ermöglicht, wobei das Feld einen Aus-
gang aufweist, der zumindest einem der folgenden zugeführt wird: der Vi-
deoanzeige, dem multifunktionalen Eingangs-/Ausgangssignalmodul (6)
und dem Speicher (3); und

ein graphisches Compiler-Programm, das in dem Speicher (3) gespeichert ist und durch den Prozessor (2) ausführbar ist bei Vollendung des Set-Up-Programms, um ein wissenschaftliches Instrumentationsemulationsprogramm zu kreieren, um eine Gesamt-Signalverarbeitungsfunktion des Feldes von ausgewählten verbundenen Instrumentationseinheiten auszuführen, wobei das graphische Compiler-Programm in der Lage ist, Signalarückkopplung in dem Feld aufzulösen und eine Sequenz zu bilden in dem Emulationsprogramm der vorbestimmten Signalverarbeitungsfunktionen der Instrumentationseinheiten, die jede auszuführen ist einmal zwischen sukzessiven Samples eines Eingangssignals, das erhalten wird unter Verwendung des multifunktionalen Eingangs-/Ausgangssignalmoduls, wobei das Emulationsprogramm Operationen aufweist, die gemäß der spezifizierten Abtastfrequenz konfiguriert sind, zum Abtasten des Eingangssignals, um einen Datenstrom von dem multifunktionalen Eingangs-/Ausgangssignalmodul (6) zu schaffen;

eine Instrumentenemulationsunteranordnung, die gekoppelt ist mit dem Prozessor (2), dem Speicher (3) und dem Videoanzeigengenerator, wobei die Instrumentenemulationsunteranordnung direkt verbunden ist mit dem multifunktionalen Eingangs-/Ausgangs-Signalmodul, wobei die Instrumentenemulationsunteranordnung aufweist:

einen zweiten Prozessor zum Ausführen des wissenschaftlichen Instrumentenemulationsprogramms, um kontinuierlich die Gesamt-Signalverarbeitungsfunktion auszuführen, derart, dass jedes Sample des Datenstroms von sukzessiven Samples verarbeitet wird während einer einzelnen Sample-Zeit, um eine Felddarstellung des wissenschaftlichen Instrumentenemulators zu produzieren, bevor ein sukzessives Sample des Eingangssignals erhalten wird durch das multifunktionale Eingangs-/Ausgangssignalmodul (6);

wobei zumindest eine einer Mehrzahl von Operationen durchgeführt wird, wobei die Mehrzahl von Operationen jede von Anzeigen der Felddarstellung auf der Videoanzeige (10) in Echtzeit, Speichern der Felddarstellung in dem

Speicher (3) und Bereitstellen der Felddausgabe zu dem Signalausgang des multifunktionalen Eingangs-/Ausgangssignalmoduls (6) in Echtzeit umfaßt.

2. Wissenschaftlicher Instrumentenemulator gemäß Anspruch 1, bei welchem die Instrumentationseinheiten, die repräsentiert sind durch ein entsprechendes Icon, welches geeignet ist, auf der Videoanzeige angezeigt zu werden während des Betriebs des Set-Up-Programms, wobei jede gewünschte Instrumentationseinheit ausgewählt werden kann, die angeordnet ist in und verbunden ist mit anderen Instrumentationseinheiten in dem Feld.

3. Wissenschaftlicher Instrumentenemulator gemäß Anspruch 2, bei welchem das Feld eine Datensignal-Rückkopplungsschleife aufweist, die zwischen einem Ausgang einer der Instrumentationseinheiten und einem Eingang einer der Instrumentationseinheiten verbunden ist.

4. Wissenschaftlicher Instrumentenemulator gemäß Anspruch 1, bei welchem das multifunktionale Eingangs-/Ausgangssignalmodul einen zusätzlichen Signalausgang aufweist zur Verbindung mit externer Hardware und welcher verfügbar ist als ein Echtzeit-elektrisches Signal, welches geeignet ist, die externe Hardware zu bedienen oder zu triggern.

5. Wissenschaftlicher Instrumentenemulator gemäß Anspruch 4, bei welchem das multifunktionale Eingangs-/Ausgangssignalmodul sowohl analoge als auch digitale Signalausgänge aufweist.

6. Wissenschaftlicher Instrumentenemulator gemäß Anspruch 4, bei welchem die Felddausgabe, die in dem Speicher gespeichert wird, verfügbar ist für nachfolgende graphische Manipulation und/oder Tabulation durch den Computer.

03.03.02

4

7. Wissenschaftlicher Instrumentenemulator gemäß Anspruch 1, bei welchem die Instrumentenemulations-Prozessoranordnung des Weiteren einen Videogenerator aufweist, der mit dem zweiten Prozessor verbunden ist, zum Bereitstellen einer anzeigbaren Repräsentation der Felddausgabe, bevor ein nachfolgendes Sample erhalten wird.
8. Verfahren zum Bestimmen der Ausführungsreihenfolge vorbestimmter Signalverarbeitungselemente in einem Computer, der eine Zentralverarbeitungseinheit (2) und einen elektronischen Speicher (3) aufweist, um in Echtzeit eine mathematische/Signalverarbeitungsfunktion eines emulierten Instruments zu implementieren, welches zumindest eine wiederholt abgetastete Signaleingabe aufweist, um davon eine Signalausgabe zu bilden, wobei das Verfahren die folgenden Schritte aufweist:
10. Anwenden eines Set-Up-Programms, welches in dem Speicher (3) gespeichert wird und durch die Zentralverarbeitungseinheit (2) ausgeführt wird, um einem Bediener zu erlauben, ausgewählte Instrumentationseinheiten miteinander zu verbinden, wobei eine Abtastfrequenz für ein multifunktionales Eingangs-/Ausgangssignalmodul (6) spezifiziert wird, wobei das Set-Up-Programm Rückkopplungsverbindung in dem Feld ermöglicht, wobei
15. das Feld einen Ausgang aufweist, welcher zumindest einem der folgenden zugeführt wird: einer Videoanzeige (10), dem multifunktionalen Eingangs-/Ausgangssignalmodul (6) und dem Speicher (3), wobei jede der Instrumentationseinheiten eine mathematische/Signalverarbeitungsfunktion aufweist;
20. Kreieren eines wissenschaftlichen Instrumentenemulationsprogramms unter Verwendung eines Compiler-Programms, welches in dem Speicher (3) gespeichert wird und durch die Zentralverarbeitungseinheit (2) bei Vollendung des Set-Up-Programms ausführbar ist, wobei das wissenschaftliche Instrumentenemulationsprogramm die Gesamt-Signalverarbeitungsfunktion des Feldes ausgewählter verbundener Instrumentationseinheiten ausführt.
25. 30.

Bestimmen einer Ausführungsreihenfolge der mathematischen/Signalverarbeitungsfunktion mit den folgenden Schritten:

Repräsentieren der Funktion als eine Sequenz elementarer mathematischer Schritte, die ihrerseits repräsentierbar sind unter Verwendung vordefinierter Signalverarbeitungselemente, und

Anordnen der vorbestimmten Signalverarbeitungselemente in der Sequenz für sequentielle Ausführung innerhalb der Zeit zwischen sukzessiven Samples und Beginnen mit der zumindest einen abgetasteten Signaleingabe,

wobei der Bestimmungsschritt hierdurch ein Feld von Instrumentationseinheiten bildet, welches die Gesamtschaltungsmathematische/Signalverarbeitungsfunktion aufweist;

Ausführen der Gesamt-Signalverarbeitungsfunktion, derart, dass jedes Sample des zusammenhängenden Datenstroms verarbeitet wird während

einer einzelnen Abtastzeit, um eine Felddausgabe der gesamtmathematischen Signalverarbeitungsfunktion zu produzieren, bevor ein sukzessives Sample des Eingangssignals erhalten wird durch das multifunktionale Eingangs-/Ausgangssignalmodul (6);

Repräsentieren der Gesamtschaltungsmathematischen/Signalverarbeitungsfunktion als eine Sequenz von Ereignissen, die darstellbar sind als vorbestimmte Signalverarbeitungselementereignisse;

Anordnen der vorbestimmten Signalverarbeitungselementereignisse in der Sequenz für sequentielle Ausführung, beginnend mit der zumindest einen abgetasteten Feldsignaleingabe, wobei eine bestimmte Rechnerzeit der vorbestimmten Signalverarbeitungselemente gesichert wird durch Ausführen jedes der vorbestimmten Signalverarbeitungselemente einmal zwischen sukzessiven Samples, wobei das Verfahren in der Lage ist, Datensignal-Rückkopplung in dem Feld von Instrumentationseinheiten aufzulösen;

Anzeigen der Felddausgabe auf der Videoanzeige (10) in Echtzeit;

00:00:00

-6-

Speichern der Feldausgabe in dem Speicher (3); und
Bereitstellen der Feldausgabe an den Signalausgang des multifunktionalen
Eingangs-/Ausgangssignalmoduls in Echtzeit.

9. Verfahren gemäß Anspruch 8, bei welchem ein Parameter des funktionalen
Blocks spezifiziert werden kann;

10. Verfahren gemäß Anspruch 8, bei welchem das Feld zumindest eine Da-
tensignal-Rückkopplungsschleife umfasst, in welcher ein Ausgang von ei-
ner der Instrumentationseinheiten verbunden ist, um einen Eingang von ei-
ner der Mehrzahl von Instrumentationseinheiten zu bilden, und ein ur-
sprüngliches Ergebnis eines entsprechenden der ausführbaren Ereignisse in
der Sequenz ausführbarer Ereignisse verwendet wird in einer wiederholten
Ausführung eines früheren Ereignisses, um ein modifiziertes Ergebnis des
entsprechenden einen der ausführbaren Ereignisse von einer vorherigen
Abtastperiode zu erzeugen.

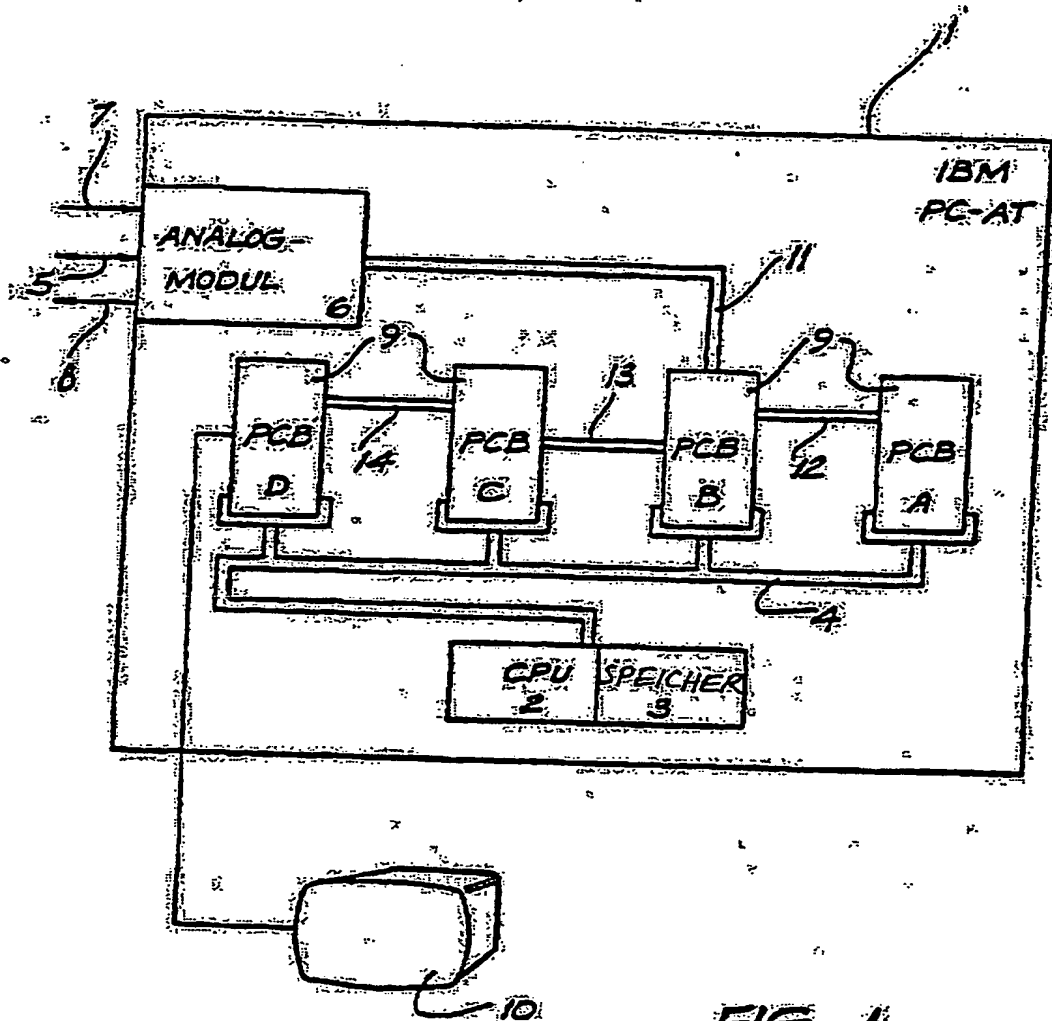


FIG. 1

06-03-02

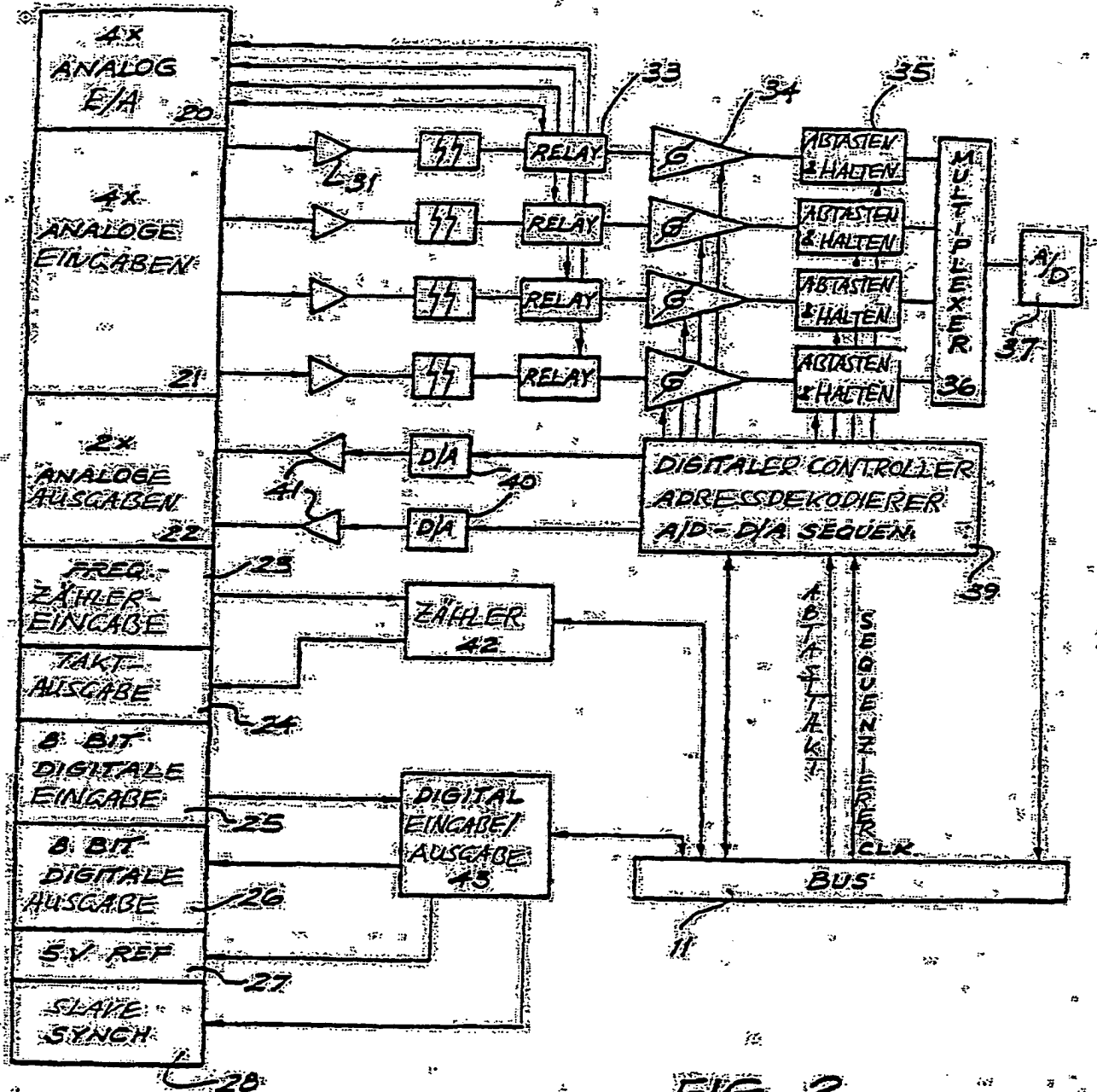


FIG. 2

00-0000

3/15

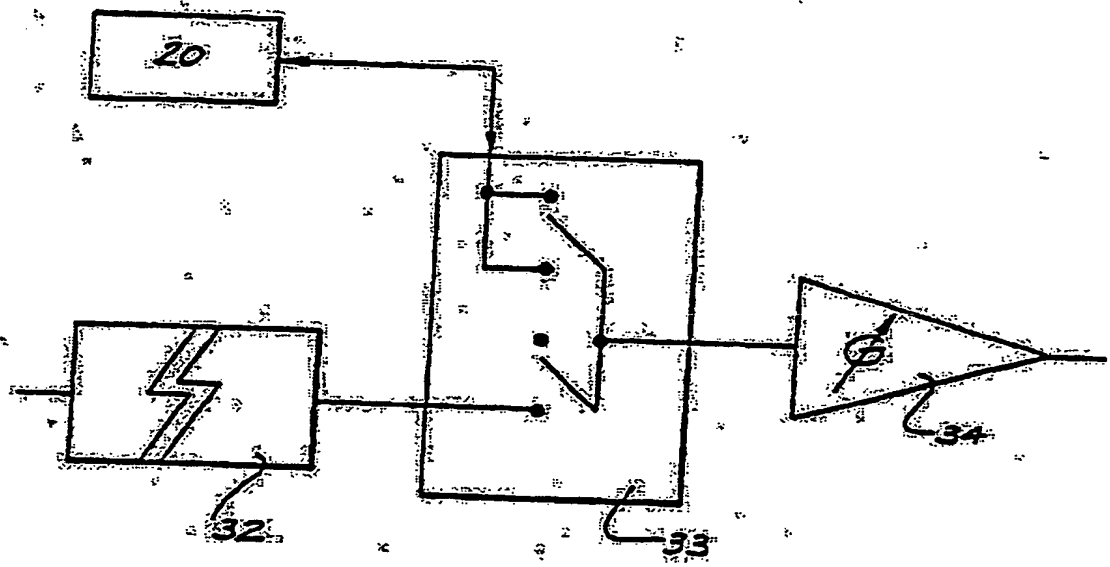


FIG. 3

SPEICHERKARTE

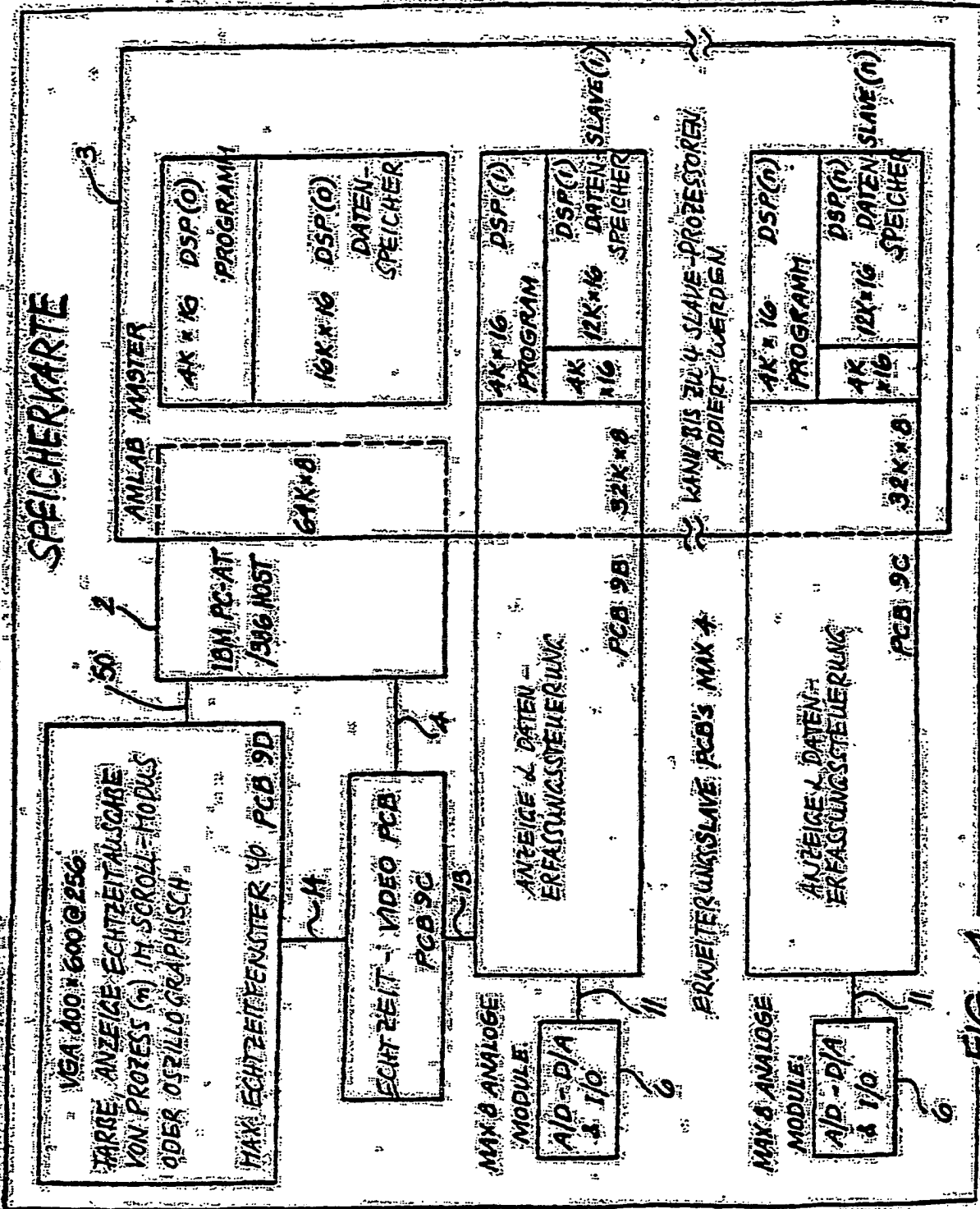
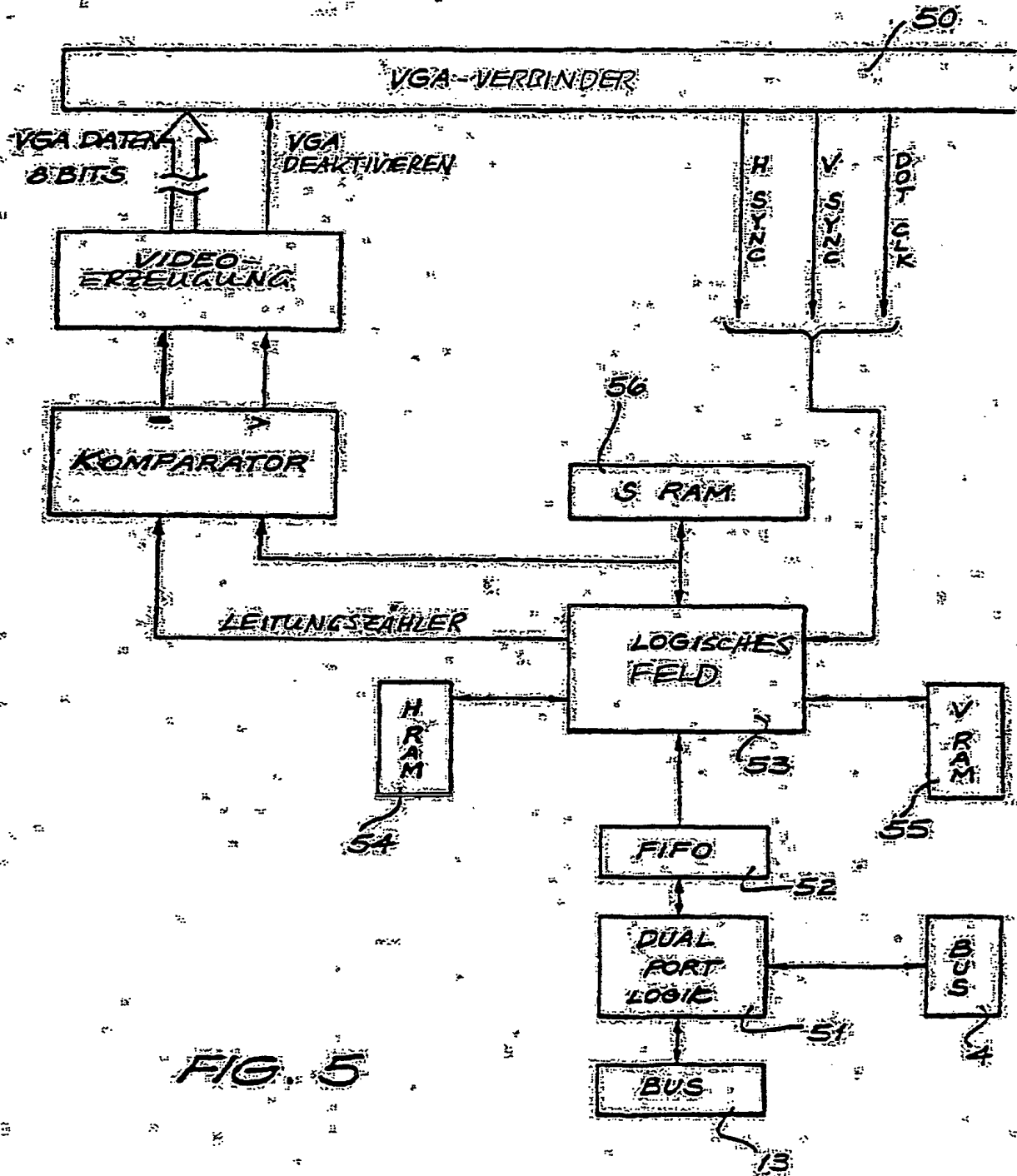


FIG. 4



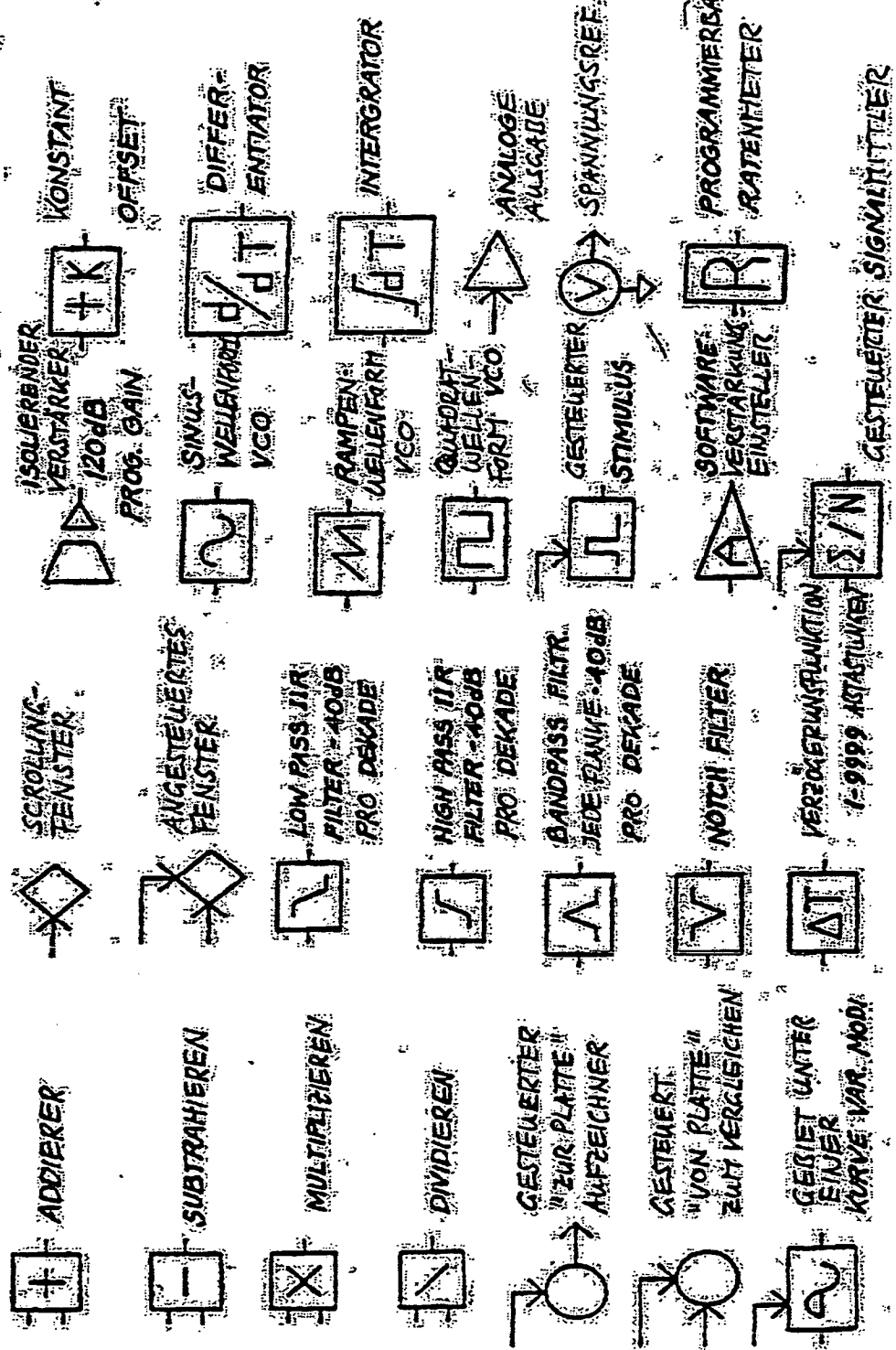


FIG. 6

00000000

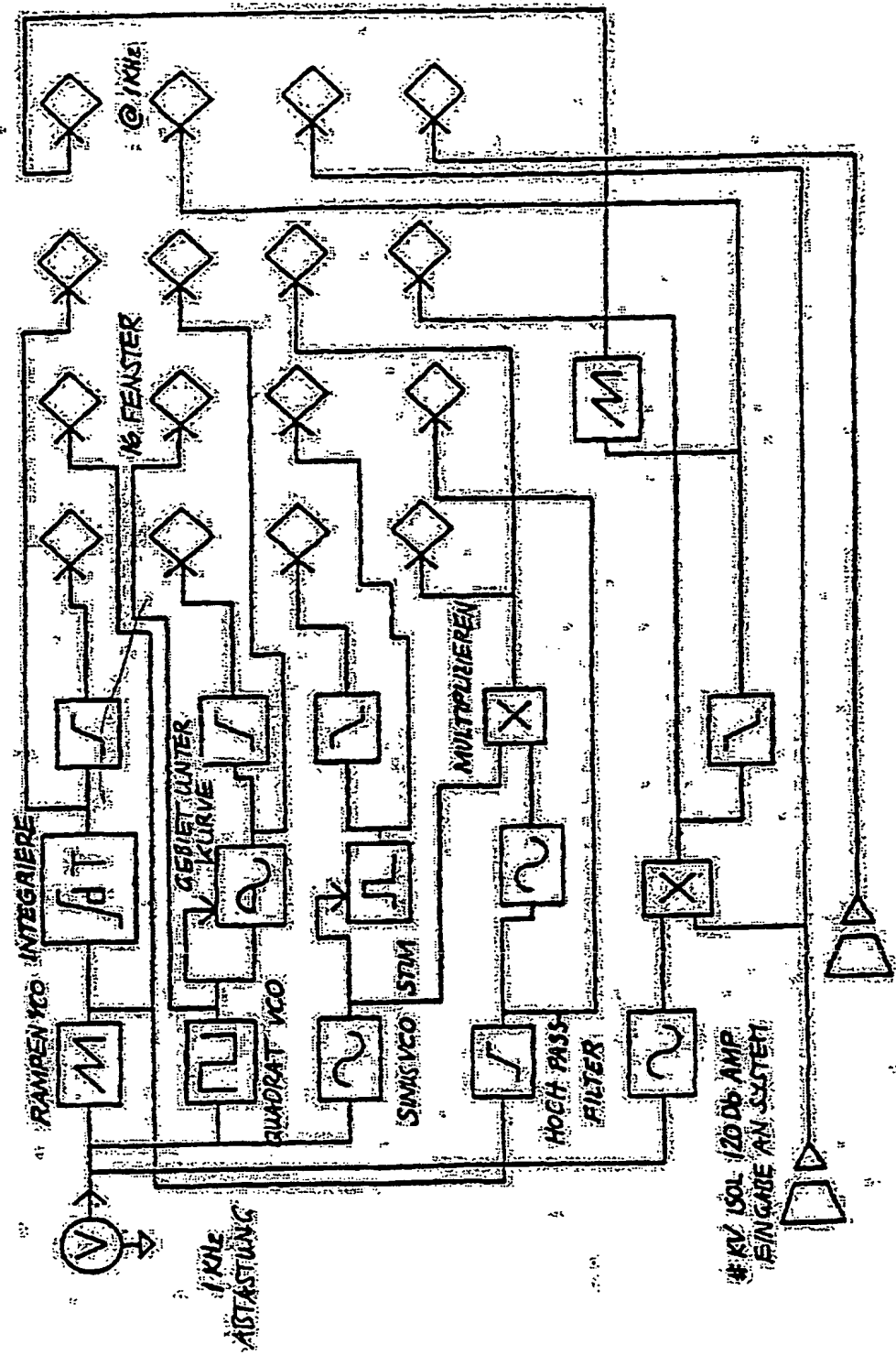


FIG. 7

8/15

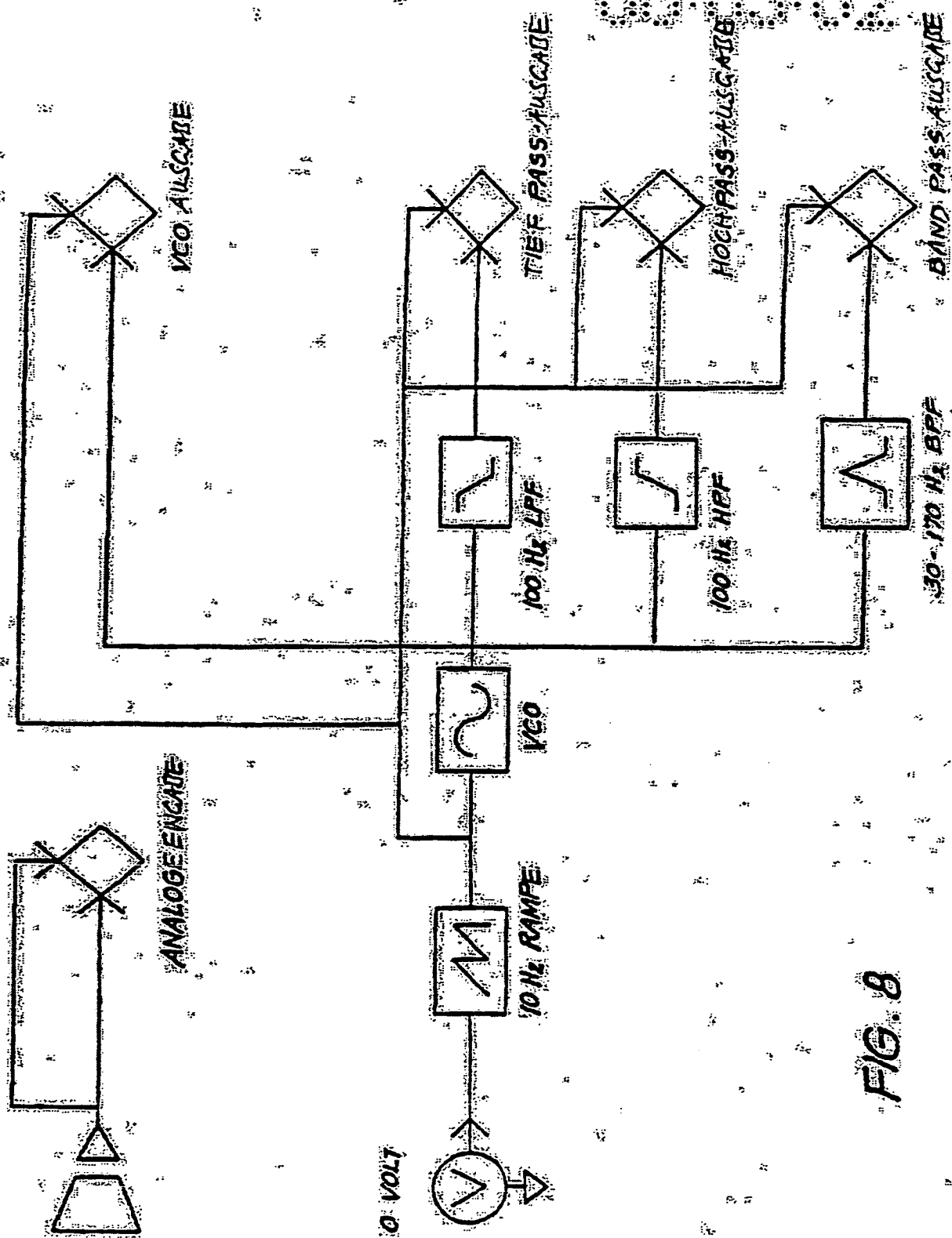


FIG. 8

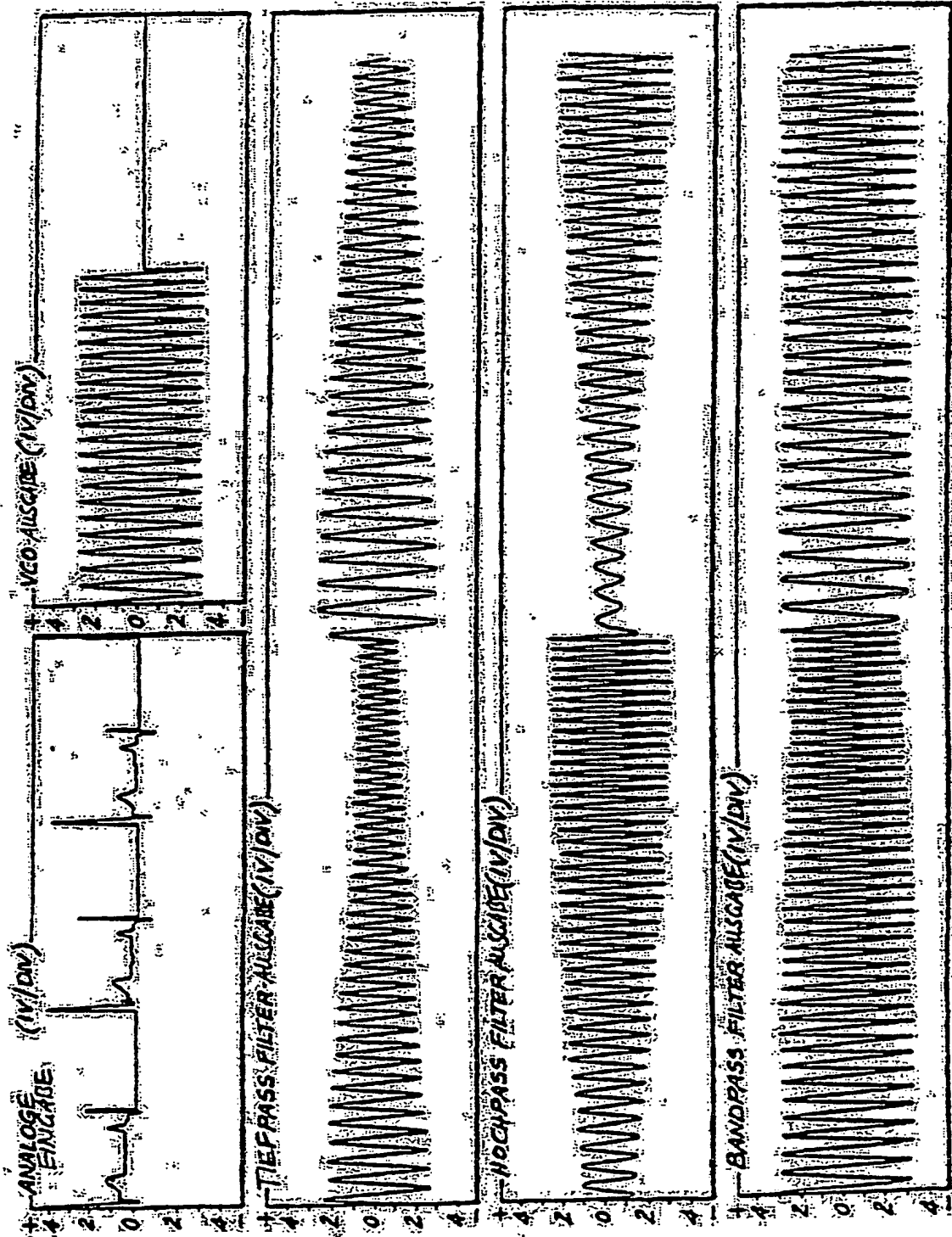


FIG. 9

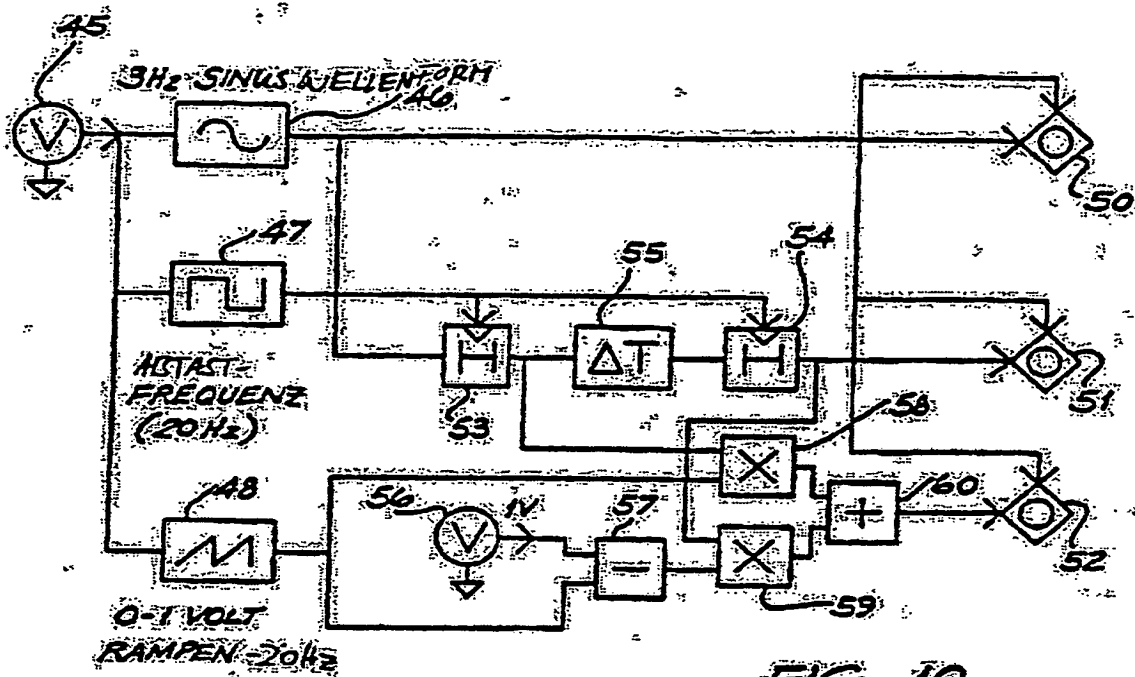


FIG. 10

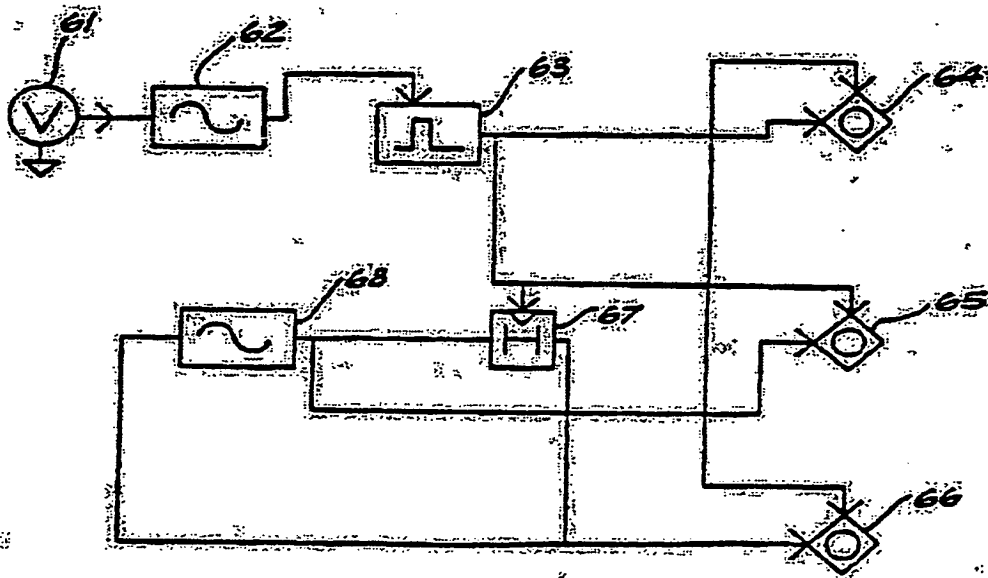


FIG. 12

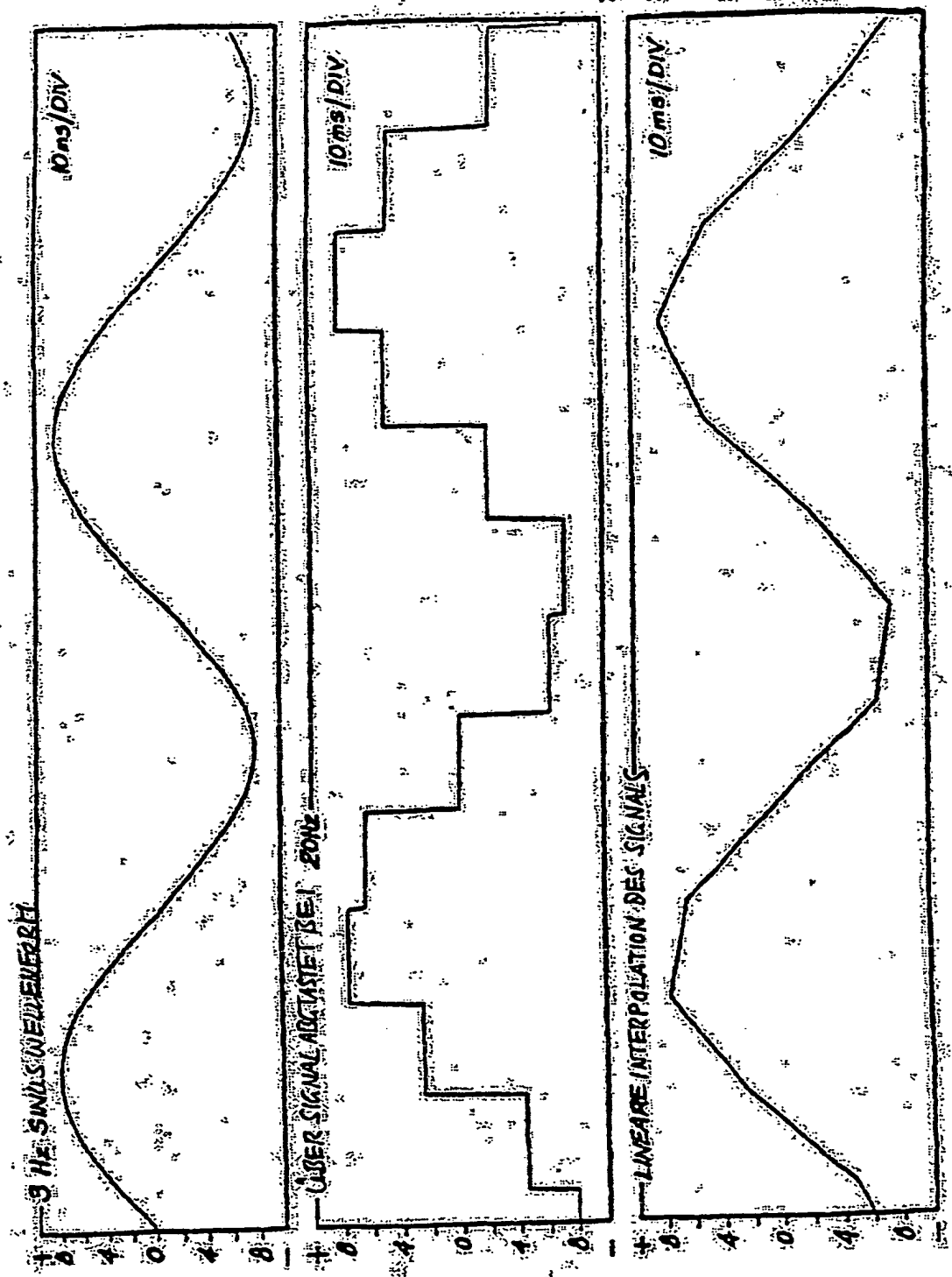


FIG. 11

12/15

000000

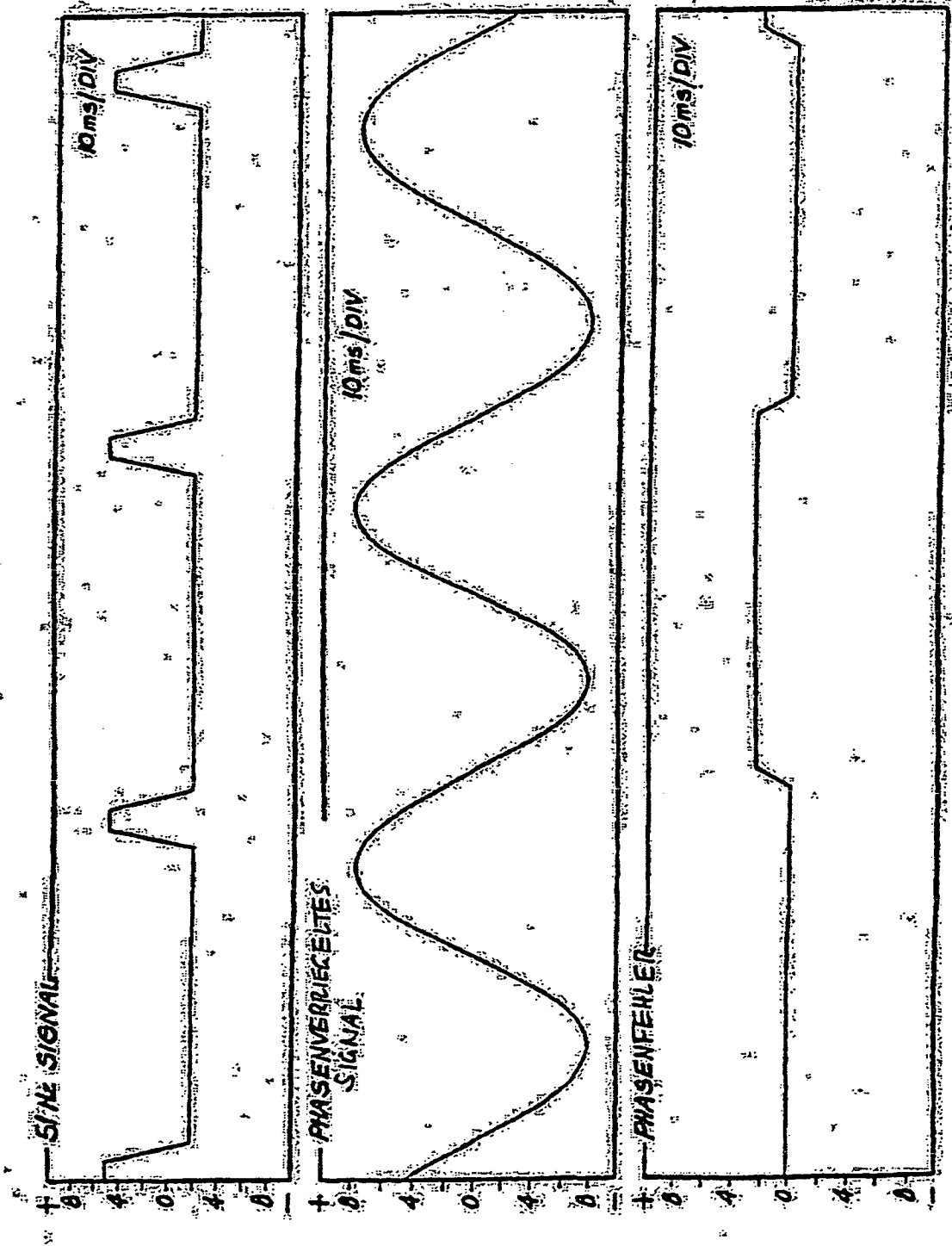


FIG. 13

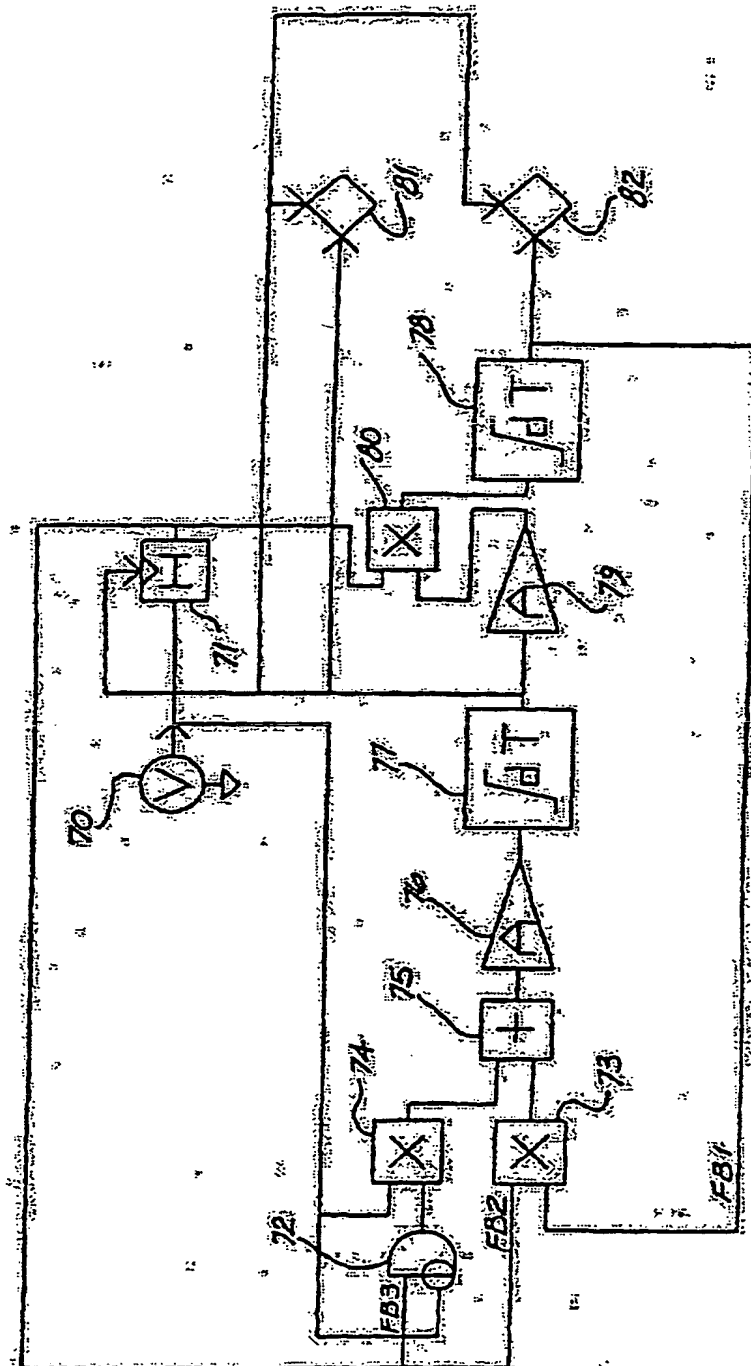


FIG. 14

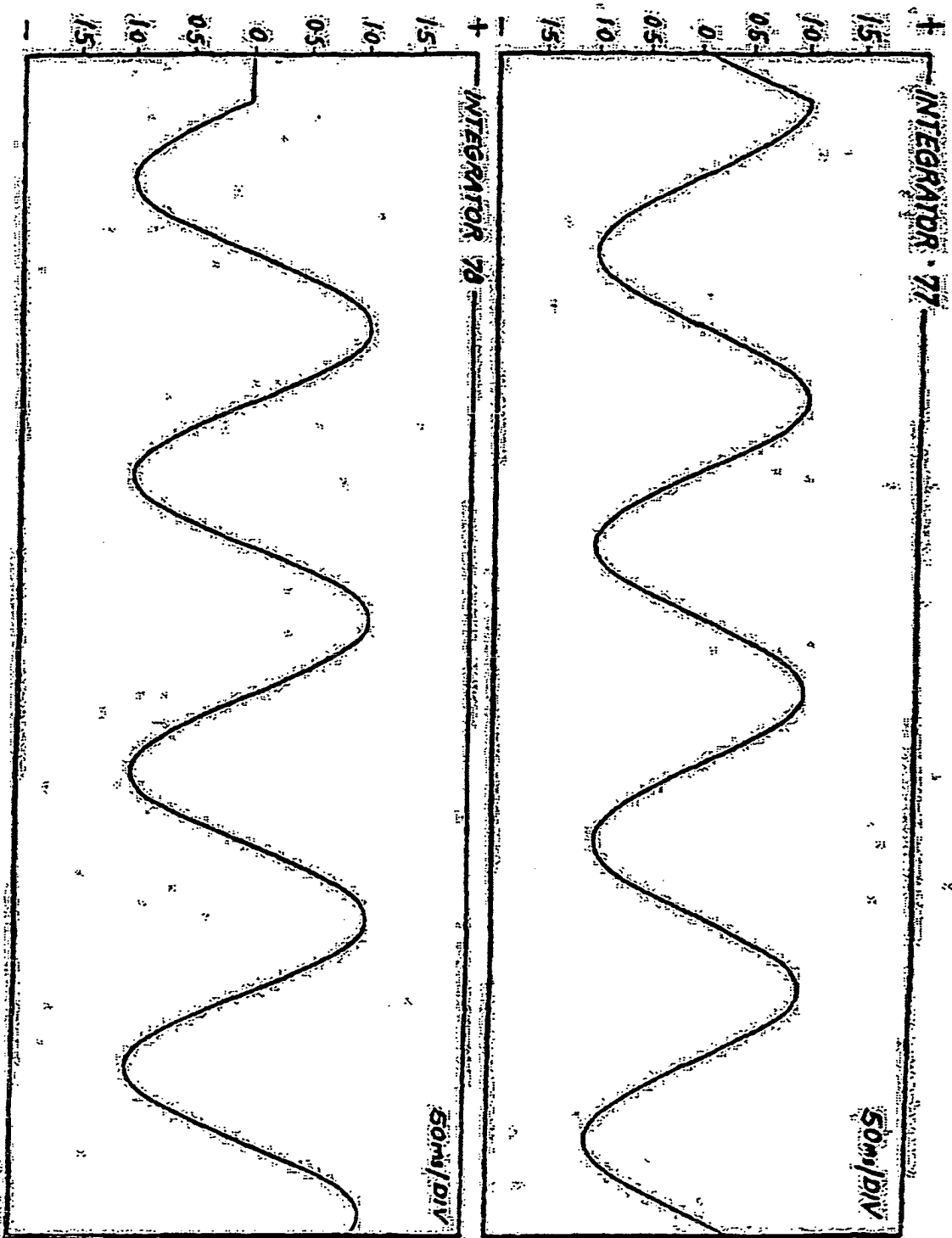


FIG. 15

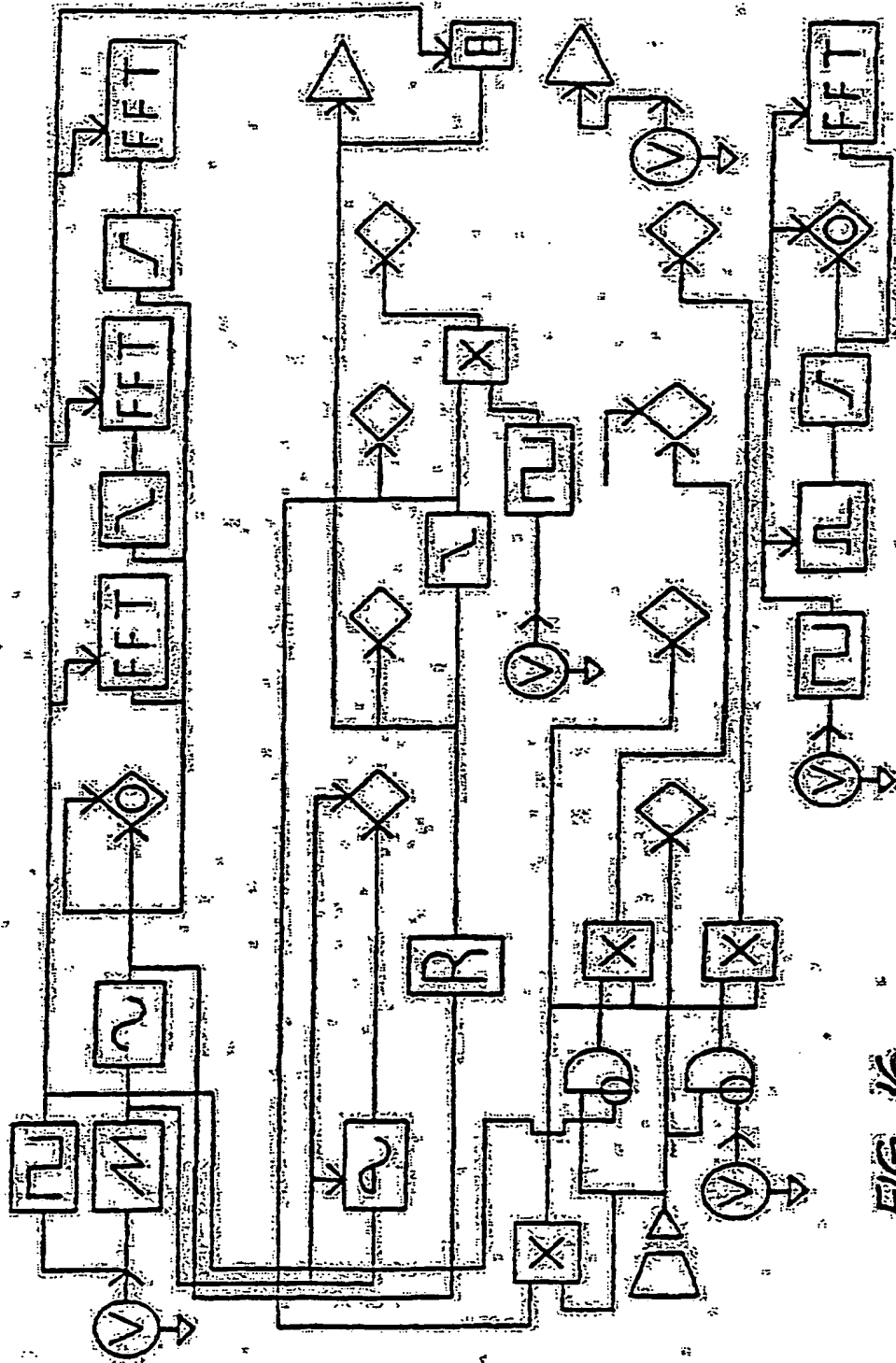


FIG. 16

THIS PAGE BLANK (USPTO)

(19)



Europäisches Patentamt
European Patent Office
Office européen des brevets



(11)

EP 0 573 503 B1

(12)

EUROPEAN PATENT SPECIFICATION

(45) Date of publication and mention
of the grant of the patent:
11.12.2002 Bulletin 2002/50

(51) Int Cl.7: **G06F 9/44**, G06F 17/40,
G01R 13/04, G01R 29/00,
G01R 13/34

(21) Application number: 92905546.5

(86) International application number:
PCT/AU92/00076

(22) Date of filing: 25.02.1992

(87) International publication number:
WO 92/015959 (17.09.1992 Gazette 1992/24)

(54) SCIENTIFIC INSTRUMENT EMULATOR

WISSENSCHAFTLICHES EMULATORGERÄT
EMULATEUR D'INSTRUMENT SCIENTIFIQUE

(84) Designated Contracting States:
AT BE CH DE ES FR GB IT LI LU NL

(30) Priority: 28.02.1991 AU 486991

(43) Date of publication of application:
15.12.1993 Bulletin 1993/50

(73) Proprietor: ASSOCIATIVE MEASUREMENT PTY.
LTD.
North ryde, NSW 2114 (AU)

(72) Inventors:

- WILLIAMS, Donald, Victor
Balmain, NSW 2041 (AU)
- KEEBLE, John, Brian
Neutral Bay, NSW 2089 (AU)
- OATES, John, David
Quakers Hill, NSW 2763 (AU)
- CAMPOS, Alejandro, Guillermo
North Rocks, NSW 2151 (AU)

(74) Representative: Altenburg, Udo, Dipl.-Phys. et al
Patent- und Rechtsanwälte
Bardehle . Pagenberg . Dost . Altenburg .
Geissler . Isenbruck
Gallieplatz 1
81679 München (DE)

(56) References cited:
EP-A- 0 351 961 US-A- 4 064 394
US-A- 4 315 315 US-A- 4 507 740
US-A- 4 868 785

- AMERICAN JOURNAL OF PHYSICS, AUG. 1989,
USA, vol. 57, no. 8, ISSN 0002-9505, pages
741-746, XP002020430 WHITELEY R V ET AL:
"An inexpensive, versatile experiment for
teaching computer-aided experimentation"
- NORTHCON. CONFERENCE RECORD,
SEATTLE, WA, USA, 9-11 OCT. 1990, 1990, LOS
ANGELES, CA, USA, ELECTRON.
CONVENTIONS MANAGE, USA, pages 353-358,
XP002020431 TREAT H: "Using mathematica in
support of LabVIEW: power in the laboratory"
- PC BUSINESS SOFTWARE, 1989, UK, vol. 14, no.
1, ISSN 0038-0652, pages 2-3, XP002020432
"LabVIEW Version 2.0"

Note: Within nine months from the publication of the mention of the grant of the European patent, any person may give notice to the European Patent Office of opposition to the European patent granted. Notice of opposition shall be filed in a written reasoned statement. It shall not be deemed to have been filed until the opposition fee has been paid. (Art. 99(1) European Patent Convention).

EP 0 573 503 B1

Description

[0001] The present invention relates to scientific instruments and, in particular, to an emulator which enables the function of various scientific instruments, when interconnected, to be emulated or replicated without the need to purchase, install, interconnect, and test, the various scientific instruments. In its preferred form, the present invention enables measurements to be taken, and the use of measurement processes, which were unable to be taken, and used, with the interconnected instruments.

BACKGROUND ART

[0002] In many fields of activity having a scientific basis, scientific instruments are used to analyse, record, and monitor the outputs of various devices. Such devices include strain gauges, electro-cardiograph (ECG) devices, microphones, and pressure, temperature, flow rate and like transducers. Accordingly, such scientific instruments are used in civil engineering, electrical engineering, acoustics, hydraulic engineering, chemical processes, bio-medical engineering and so on.

[0003] A wide range of such scientific instruments are generally required in order to undertake desired measurements. Such instruments include generators for various wave-forms (such as sine, square, ramp, and triangle); signal processing devices such as differentiators, integrators, filters, multipliers, and so on; analysers such as that required to carry out the Fast Fourier Transform, and various recording devices such as a chart recorder, a data logger, a cathode ray oscilloscope or a transient recorder.

[0004] Such instruments or devices are each relatively expensive and thus any research institution, or like organisation only has a limited number of such devices. Accordingly, there is considerable competition amongst persons or groups within such organisations who wish to utilise the devices. Once the devices have been obtained in order to carry out the intended operation, it is necessary for the devices to be located together, interconnected, and tested in order to ensure that the interconnections are correct. Only once this procedure has been carried out is it then possible to commence the intended operation.

[0005] It is known in the art to provide instruments which are essentially highly advanced cathode ray oscilloscopes. One such instrument is sold under the name SUPERSCOPE by G W Instruments of Summerville Ma USA 02143 and enables waveforms captured by what is essentially a storage oscilloscope to also be displayed on an APPLE (Registered Trade Mark) MacIntosh (Registered Trade Mark) Computer. However, this device does not appear to emulate instruments, rather it only captures and stores for subsequent display the output of such instruments.

[0006] It is also known to utilise software for data acquisition. One such program offered by LABTECH of Wilmington, Ma and San Francisco, California, USA is a graphical interface which again accepts signals from various hardware items external to the computer. The software collects data from multiple channels, carries out an analysis and, if necessary a reduction of the data and produces displays. This activity is available in real-time. Again, no attempt is made to emulate the function of scientific instruments. A further difference is that the graphical interface connects to a number of different machines, rather than being essentially integrated into one unit.

[0007] A data acquisition and analysis system sold under the trade name LABVIEW2 by National Instruments of Austin, Texas, USA is also known. In this system, icons are used to represent both data acquisition functions and data analysis functions. These icons are used to generate executable code which is only executed by the personal computer into which the software of the system is loaded. This system uses no hardware, only software. The data acquisition and analysis functions can be sequentially linked so that data input into the computer can be first acquired and then analysed. Finally the analysed data is presented. It is to be noted that the computer functions to make desired interconnections between external instruments but does not emulate instruments or compile a program to replicate the signal processing functions of an array of inter-connected instruments. For example, no provision for feedback from one icon back to another in the control sense, is made. The slow processing speeds, and the indeterminate nature of the multi-tasking environment of the APPLE (Registered Trade Mark) personal computer, require the provision of a time "co-ordinate" to accompany the acquired data.

[0008] This system is a virtual instrument which is to say it is characterised by software simulation only and is to be contrasted with the new concept of emulation. Emulation involves flexible, multipurpose, re-programmable hardware which executes code derived from a graphic compiler at a fast operating speed which enables real time instrument emulation.

[0009] US 4,315,315 relates to a process for automatically producing a computer program in machine assembly language directly from a two-dimensional network representing the flow of data and control logic which it is desired to accomplish on a specified general purpose digital computer. However, emulation is not performed.

[0010] The article "Using Mathematica in Support of LabView: Power in the Laboratory", NORTHCON Conf. Rec. Seattle, WA, USA, 9-11, Oct. 1990, pp. 353.358, relates to a graphics oriented computer language for creating programs which are executed on personal computer. There is no emulation performed.

[0011] Therefore, the object of the present invention is to provide a scientific instrument emulator and a corresponding method which reproduce the function of scientific instruments. This object is achieved by the emulator according to claim 1 and by the method according to claim 8, respectively. Further embodiments are defined in the dependent claims.

[0012] The task of the emulator is to reproduce by means of a combination of both hardware and software, the functions of various scientific instruments thereby enabling not only the individual functions of the instruments to be replicated, but for such replicated functions to be interconnected with each other so that the emulator as a whole can replicate the overall function of an array formed by interconnected scientific instruments.

[0013] A number of very substantial advantages are attained thereby. Firstly, the overall cost of the apparatus is substantially reduced since the individual scientific instruments the functions of which are to be replicated, need not be purchased. Secondly, a substantial saving in set up time is also achieved since it is not necessary for such instruments to be bought from various locations and assembled at the one point and physically interconnected with wiring, plugs, adaptors, and so on. Thirdly, the results achieved are much improved in that the output of the array of interconnected scientific instruments is not only able to be displayed in real time, but it is, in addition, processed in real time and is also able to be stored for subsequent manipulation, analysis and evaluation. Finally, the results of simultaneous or parallel measurements can be more easily associated so that the interrelationships between measurements can be ascertained, particularly in complex systems.

[0014] The invention is set out in appended apparatus claim 1 and method claim 8.

[0015] Preferably the instrumentation units are each represented by a corresponding icon which is able to be displayed on the video display screen during the operation of the set-up program to enable the operator to select, and locate in the array, the desired instrumentation unit.

[0016] The compiler generates in the memory a number of different programs representing the array of instrumentation units. These programs are distributed to the processor in the computer system to emulate the desired array of instrumentation units. Preferably, the compiler generated programs are stored in memory to create preconfigured instrumentation arrays instantly capable of executing desired processing functions. These programs store the accumulated intellectual product of the user.

[0017] Preferably the number of processors, video generators and analog signal modules is expandable to obtain the desired degree of complexity and/or capability of the array of instrumentation units.

[0018] The analogue signal module has available a real time electric signal which can be used to operate, or trigger, other items of hardware. Preferably it also provides both analogue and digital output.

[0019] The input/output from the scientific instrument emulator is preferably also available from other resources available within the computer including network communications interfaces (RS232, ETHERNET etc) and bus interfaces such as IEEE-488-GPIB, ISA and EISA. These resources can be provided independently by the purchaser or original equipment manufacturers (OEM's).

[0020] The data stored in the memory means is also preferably available for data "export" to various standard computer packages such as those sold under the trade marks EXCEL, LOTUS and AXUM whereby the data accumulated can be manipulated for subsequent graphical presentation and tabulation to facilitate report generation.

[0021] According to a second aspect of the present invention there is disclosed a scientific instrument for measuring and recording electrical waveforms, said instrument comprising a computer having a central processing unit and electronic memory means positioned within a housing, said housing including provision for at least one floppy disc nacelle; and a plurality of electrical connectors accommodated in the space allocated for said one floppy disc nacelle, said electrical connectors being connected to said memory means.

[0022] According to a third aspect of the present invention there is disclosed a compilation method for generating object code to implement the mathematical/signal processing function of an electrical circuit functional block having at least one input to form an output thereof, said method comprising the steps of representing said function as a sequence of elemental mathematical steps each of which is itself directly representable in said object code, and arranging the object code steps in said sequence for sequential execution commencing with said input(s). Preferably a parameter of said functional block is able to be specified. Also disclosed is a compilation method for generating object code to implement the mathematical/signal processing function of an electrical circuit having at least one input and at least one output and formed by interconnection of a plurality of functional blocks each of which has a mathematical/signal processing function for which an object code has been compiled in accordance with the above, said method comprising the steps of representing the electrical circuit mathematical/signal processing function as a sequence of events each of which is representable in said object code, and arranging said object code events in said sequence for sequential execution commencing with said input(s).

DESCRIPTION OF THE DRAWINGS

[0023] An embodiment of the present invention will now be described with reference to the drawings in which:

Fig. 1 is a schematic block diagram of the hardware modifications required to an IBM PC,
 Fig. 2 is a block diagram of the analogue module of Fig. 1,
 Fig. 3 is a circuit diagram of one of the identical relays of Fig. 2,
 Fig. 4 is a schematic map of the memory arrangement within the computer,
 Fig. 5 is a block diagram of the real time video printed circuit board of Fig. 4,
 Fig. 6 is a screen display listing the icons representing the various instrumentation units stored within the library program,
 Fig. 7 is a typical array formed by inter-connection of the various instrumentation units,
 Fig. 8 is a block diagram of a relatively straight forward array useful in bio-medical applications,
 Fig. 9 is a reproduction of the screen display windows corresponding to the array of Fig. 8.
 Fig. 10 is an array which forms a linear interpolation,
 Fig. 11. shows the output of the three screen displays illustrated in Fig. 10.
 Fig. 12 is an array which forms a phase locked loop,
 Fig. 13 shows the output of the three screen displays illustrated in Fig. 12,
 Fig. 14 is an array which provides an analogue solution to a second order differential equation,
 Fig. 15 shows the output of the two screen displays illustrated in Fig. 14, and
 Fig. 16 is a more complex array.

[0024] Appendices I-IV list various program fragments described hereafter.

DESCRIPTION OF THE PREFERRED EMBODIMENT

[0025] Turning now to Fig. 1, the apparatus of the preferred embodiment is able to be totally enclosed within the case 1 of a conventional IBM (Registered Trade Mark) Personal Computer having an ISA or EISA bus based on the original IBM AT. Located within the case 1 are the usual components of a central processing unit (CPU) 2, memory 3 and 8MHz bus 4.

[0026] Located within a spare 5¹/₄ inch floppy disc nacelle is an analogue module 6 onto which are mounted dedicated input plugs 7, dedicated output plugs 8, and general input/output lines 5 for amplifiers, frequency counters, sample clock synchronising, digital inputs and the like.

[0027] Located on the bus 4 are four slots for printed circuit boards 9. The four PC boards 9 are indicated A, B, C and D respectively. The three adjacent PCB's A to C inclusive are respectively a master PCB, a slave PCB and a video PCB. The video PCB in turn drives a known VGA printed circuit board D sold by Tseng Labs which can provide 800 x 600 resolution in 256 colours. This PC board D is directly connected to the video display screen 10.

[0028] Within the analogue module 6 are located the following system resources:

1 FOUR isolated (optional) analog i/p channels. Each channel has programmable 9-120dB gain (3 micro volts resolution @ signal to noise ratio of one), programmable anti-alias filtering and an ADC conversion of 12 bit resolution. Each channel can be AC or DC coupled with long AC coupling time constants (2 minutes) and has independent controls of AC or DC offsets which can be controlled from the runtime screens. The sampling rate can be 15 KHz per channel (depending on the project processing load) and the number of analog modules attached to the same slave processor card. The inputs are isolated to 3.5KV continuous.

2 TWO analog outputs with a voltage range of +/-10 Volts and a current capacity of +/- 100mA. These can be used for strain gauge biasing (AC or DC driven), control outputs etc.

3 FOUR selectable high level analogue outputs, one from each of the amplifiers above. These drive digital FM tape recorders to store rarely occurring events for replay into the processor (2).

OR

FOUR selectable high level inputs to each of the amplifier channels above. The system is switched into this mode for replay of events captured in output mode on tape.

4 EIGHT bits of ground referenced digital input.

5 EIGHT bits of digital output which can be used for relay drivers or event indicators.

6 ONE frequency generator output (clock generator) 0-2 MHz 0.1% accuracy.

7 ONE event counter/frequency counter. Input 0.1Hz-8MHz.

8 ONE 5 Volt reference 100mA +/- 5%. (For strain gauges etc.)

9 ONE sample clock output reference line for synchronising sampling between multiple scientific instrument emulators of the preferred embodiment.

10 ONE sample clock input reference line for synchronising sampling from a "master" scientific instrument emulator (for use with "slave" emulators), and

11 FIVE ground wires.

[0029] The analogue module 6 and PC boards 9 are each interconnected by means of different subsidiary buses 11, 12, 13 and 14 respectively.

[0030] As seen in Fig. 2, the analogue module 6 of Fig. 1 is provided with four analogue input/output connectors 20, four analogue inputs 21, two analogue outputs 22, a frequency output counter input 23, a clock output 24, an 8 bit digital input 25, an 8 bit digital output 26, a five volt reference voltage 27 and a slave synchronizing output 28.

[0031] Each of the analogue inputs 21 is connected via a front end amplifier 31 to an isolator 32, the output of which is connected to a relay 33. The relay 33 is also connected to the analogue input/output connectors 20 and to an amplifier 34 which has programmable gain. AC/DC coupling, provision for an AC corner and DC offset. The output of the amplifier 34 is in turn passed to a sample and hold circuit 35 the output of which is received by an analogue multiplexer 36. The output of the multiplexer 36 is passed via an A/D converter 37 to the subsidiary bus 11 which connects the analogue modules 6 and the PC board 9B.

[0032] The operation of the amplifiers 34 and the sample and hold circuits 35 is controlled by a digital controller, address decoder and A/D-D/A sequencer 39 which receives both data from the bus 11 and also sample clock and sequencer clock signals. The controller/decoder/sequencer 39 also outputs via D/A converters 40 to the analogue outputs 22 via an output amplifier 41.

[0033] The frequency counter input 23 and clock output 24 respectively directly communicate with a counter 42 which again communicates directly with the subsidiary bus 11.

[0034] Each of the digital inputs 25, digital outputs 26, reference voltage 27 and slave synchronizing output 28 is connected to a digital input/output circuit 43 which is in turn directly connected to the subsidiary bus 11.

[0035] The timing arrangements of the circuit illustrated in Fig. 2 are divided into two sequences. The first sequence concerns the digital input and output. When required by the program, this digital input and output is effected by individual commands from a substantially conventional data acquisition controller which forms part of the slave processor on PCB 9B (Fig. 1 and Fig. 4).

[0036] The second sequence is the flow of digital data converted from the analogue inputs, or to be converted to provide the analog outputs. This digital data is received and despatched under the control of the controller/decoder/sequencer 39 which can be preset to operate the required number of incoming and outgoing analogue channels. The controller/decoder/sequencer 39 performs one complete cycle of inputting and outputting, or sequence, every sample period and does so with minimal processor involvement, thus increasing the speed of operation of the data acquisition controller referred to above on the slave processor of PCB 9B.

[0037] Other functions of the circuit of Fig. 2, such as the frequency to be output as the clock output 24, the "range" of the frequency to be counted by the frequency counter input 23, and any synchronisation signal required for the slave synchronising signal 28, are set up at the start of the execution of the graphical compiler program by appropriately specifying the corresponding icon.

[0038] Fig. 3 illustrates in detail the nature of the relay 33 which can be switched between two positions. In the position shown, the input from the analogue input/output 20 is passed to the amplifier 34, thereby isolating the analogue inputs 21. In the alternative position, the input from the analogue inputs 21 is passed to the amplifier 34 but is also made available as an output at the analogue input/output 20.

[0039] Fig. 4 is a pictorial representation of the processor architecture and includes a memory map. The host computer 2 with its associated memory 3 is connected to the video PCB 9C which is in turn connected to the slave PCB 9B.

[0040] As indicated in Fig. 4, up to 8 analogue modules 6 are able to be connected to each PCB 9B and up to four slave PCB's 9B can be added.

[0041] The memory map is essentially three dimensional with the memory 3 of the host computer 2 overlying, and thereby being accessible to, the other memory utilising portions of the circuit.

[0042] A block diagram of the video PCB 9C is illustrated in Fig. 5. In this diagram the interconnections between the conventional video graphics adaptor (VGA) of the host computer are made via the conventional VGA connector 50. The connections between the host computer 2 and the video PCB 9C are via bus 4. Similarly the connections between the video PCB 9C and each of the slave PCB's 9B is via bus 13 as previously indicated in Fig. 1.

[0043] Data received from either bus 4 and/or 13 is passed via a dual port logic circuit 51 to a first in first out (FIFO) buffer 52. The FIFO 52 outputs to a logic array circuit 53 which has three separate random access memories, horizontal RAM 54, vertical RAM 55 and a static RAM 56.

[0044] The output of the logic array 53 is passed via comparator 57 to a video generator 58 and thence to the VGA connector 50. In addition, the logic array 53 also receives three signals from the VGA connector in the form of horizontal synchronizing pulses, vertical synchronizing pulses and a dot clock.

[0045] Essentially the logic array 53 takes the data supplied via buses 4 and/or 13 and calculates pixels to replace the individual pixels generated by the video graphics adaptor (VGA) 12 and complies with multimedia standards for communication of the video image on bus 14.

[0046] The memory 3 (Figs. 1 and 4) has a library of instrumentation units. Each of these is represented by an icon and Fig. 6 provides an indication of the range of instrumentation units able to be selected from a given library. By use

of a mouse in known fashion, the operator is able to interconnect selected instrumentation units from the icon list of Fig. 6 so as to form an array of interconnected instrumentation units as illustrated in Fig. 7. During the course of the creation of the array of Fig. 7, under software control the machine checks that the array does not have any unconnected inputs, logically unacceptable connections, and like defects. Any detected defect is indicated.

5 [0047] Once the array has been interconnected to the satisfaction both of the operator and the set-up program used during this phase, a compiler program is then run which compiles from the graphical representation of the array the executable object code which executes the overall signal processing function for the entire array. As a consequence, when, in real time, the input signal is applied to the array, the incoming signal(s) is/are manipulated and the one or more outputs of the array are indicated in real time on the video windows able to be displayed on the screen 10, stored to disc, and so on.

10 [0048] Essential to an understanding of the way in which the compiler program operates is an appreciation that each icon is itself a mini array which can be built up from very fundamental steps which are themselves easily executed by the program. For example, if the basic steps are taken to be addition and subtraction, then multiplication can be regarded as repeated addition, and division can be regarded as repeated subtraction. With this background in mind, it can be appreciated that a ramp voltage generator can be created from a zero initial level by the successive additions of very small increments until a predetermined level is reached. Then that level is itself subtracted so as to re-create the zero starting point. Then the small additions are again made, and so on.

15 [0049] A consideration of the various icons represented in Fig. 3 will also lead to the appreciation that the icon is itself essentially a graphical form of any desired shape designed to convey a representation of a specific mathematical or signal processing function which is both seen by the user and identically understood by the graphical compiler program. The shape contains at least one input node and/or at least one output node these being respectively the entry and exit points for data streams. For example, an adder having input nodes A and B and output node C is compiled to give the program $C = A + B$. Those icon elements having only one or more output nodes are termed source icon elements (eg a voltage generator) whilst those icons having only one or more input nodes are termed a destination icon (eg a display window).

20 [0050] Interconnecting any output node and terminating at an input node is a connecting path termed a data stream. The data stream carries the data type associated with the output node and, as a consequence, the input node at which it terminates must be of the same data type. This provides an additional rule inherent within the compiler program. Since the interconnections between nodes represent data streams, rather than physical wiring, it is permissible for the interconnections between icons to cross over other interconnections or even other icons without ill effect. This is because the data stream has an output node and an input node and is unconcerned with any intermediate location. The co-ordinates of the source and destination nodes on an arbitrary co-ordinate system are used to create the software "pointers" to source and destination data buffers for the execution of the signal processing functions on the data "pointed" to.

25 [0051] Also included with an icon is an icon specification where the icon itself is insufficient to describe the entire function of the icon. For example, an amplifier has a function, $\text{output} = G \times \text{input}$, where G is the gain of the amplifier. However, in order to enable the gain to be specified by the user, the gain is able to be input as a predetermined parameter by means of the icon specification.

30 [0052] It will be seen that using the above methodology, a library of prepared icons each with its own program to carry out the mathematical/signal processing function of the icon can be prepared. Furthermore, the same methodology is again applied once it is desired to compile a program to carry out the mathematical/signal processing functions of an array formed from inter-connected icons. To take a simple example of an array, consider an array having two inputs A and B and an output C, the array consisting of an adder having inputs A and B, the output of the adder being connected to an amplifier having a gain G with the output of the amplifier constituting the output of the array. The mathematical/signal processing function is $C = G \times (A + B)$. The graphic compiler of the preferred embodiment creates machine code in the same way as would a FORTRAN compiler when it was presented with substantially the same statement but present in source code written in the FORTRAN language.

35 [0053] Appendix I constitutes a program fragment of code which permits the user to decide what the icon specification will be for the given icon. The "pop-up" window portion of the display into which the user inputs the numerical value(s) to be specified is termed a "sheet".

40 [0054] Similarly the program fragment of Appendix II is an example of the programming used in compiling an icon. The code used in compiling an array is analogous.

45 [0055] Appendix III is a code fragment which is an example of the digital signal processing (DSP) implementation code. This code enables digital signals which replicate the output of actual electronic devices (as represented by an icon or array) to be created.

50 [0056] Finally, Appendix IV is a code fragment of the timing or sequencing used in the digital signal processing. This ensures the timely completion of the calculations in the time between successive samples.

[0057] The operation in "real time" is assisted by the way in which analogue input signals are acquired. The permiss-

able ± 10 V input range is represented by a 12 bit number and the analogue input signal is sampled at a sampling frequency of from 1 to 60,000 points or samples per second. The exact sampling frequency is specified or selectable by the user. As a consequence of this sampling regime, all the programming steps required to emulate the particular function of an icon or an array are required to be completed, and thus create an output, in the time available between sampling points. In this way, the output for a given sampling point is calculated and hence created, prior to the receipt of the information for the next sampling point. This procedure enables real time operation with the proviso that if the calculation time exceeds the sampling period then either additional computer resources in the form of extra processor and/or memory must be provided to reduce the calculation time, or the sampling frequency must be reduced, thereby increasing the time for calculation. The increasing processing power of modern computers means that, in practice, any limitation to the sampling frequency is not of practical concern.

[0058] It will also be appreciated in connection with the above that the calculation to be performed by the computer can include feedback of an output at one sample time which then constitutes an the input for a subsequent calculation at the next sample time. The subsequent calculation must, however, be completed within the sampling period.

[0059] Fig. 8 illustrates two relatively simple arrays. The input and output waveforms to, and from, those arrays are respectively illustrated in Fig. 9. It will be seen that the input and output of the first array comprises an analogue voltage from an ECG. For the other array, the four outputs comprise the output of the voltage controlled oscillator, and that output when respectively passed through a low pass filter, a high pass filter and a band pass filter. Note that since the arrays are not in any way related, although all the waveforms can be simultaneously generated and/or displayed, it is not necessary for the results to be linked in any way.

[0060] Turning now to Fig. 10, this drawing illustrates the array able to be first drawn and then compiled to emulate a linear interpolation circuit. A constant voltage source 45 is applied as an input to three voltage controlled oscillators 46-48 which respectively have a sine-wave output, a square-wave output and a ramp output. The output of the sine-wave is set at 3Hz whilst the output of the other two generators 47 and 48 is set at the sampling frequency of 20Hz. The output of the sine-wave oscillator 46 is used to trigger each of three displays 50-52. The output of the square-wave oscillator 47 is used to trigger two sample and hold circuits 53, 54 which are separated by a time delay circuit 55 having a delay which is approximately equal to the period of the sampling. This ensures that the outputs of the two sample and hold circuits 53, 54 represent the results of successive samples.

[0061] A further voltage reference 56 is set at one volt and comprises one input to a subtractor 57. The linear interpolation is carried out by the two multipliers 58, 59 and the adder 60. The multipliers multiply the ramp gradient by the correct proportion determined by the amplitude differences of successive samples held in circuits 53, 54.

[0062] The "input" sine-wave at 3Hz which constitutes the screen display 50 is illustrated in Fig. 11 as is the sampled sine-wave which constitutes the display 51. The linear interpolation created from the sampled sine-wave constitutes the display 52 and is also illustrated in Fig. 11.

[0063] Fig. 12 illustrates an array which constitutes a phase locked loop. Again, a voltage reference 61 constitutes the input of a voltage controlled sine-wave oscillator 62, the reference voltage of 61 being set so as to set the frequency produced by the oscillator 62 at 51Hz. The output of the oscillators 62 is used to trigger a pulse circuit 63. The output of the pulse circuit in turn triggers three displays 64-66 and constitutes the display signal for display 64.

[0064] In addition, the output of the pulse circuit 63 is used to trigger a sample and hold circuit 67 which has an input formed from the output of a voltage controlled sine-wave oscillator 68 which is set to have a centre frequency of 50Hz. The input to the voltage control oscillator 68 constitutes the output of the sample and hold circuit 67 which essentially represents the phase difference between the signals from the oscillators 68 and pulse circuit 63. This error signal is displayed by display 66, the output of the voltage control oscillator 68 being displayed by display 65. The three displays 64, 65 and 66 of Fig. 12 are respectively illustrated in Fig. 13.

[0065] An array to solve a second order differential equation is illustrated in Fig. 14. Here three feedback loops FB1 - FB3 are provided. In order to set the initial conditions, a voltage reference 70 set to one volt is used to provide an input to a sample and hold circuit 71 and the inverting input of a comparator 72. The output of the sample and hold circuit 71 is fed back to the comparator 72 and also to one of two multipliers 73, 74. The output of the multipliers 73, 74 are added together in adder 75 and multiplied by the gain of amplifier 76 before being integrated by the first of two integrators 77, 78. A further amplifier 79 and multiplier 80 complete the circuit. The outputs of each of the integrators 77 and 78 form the waveforms displayed by two displays 81 and 82 which are both triggered by the output of the first integrator 77.

[0066] The results are illustrated on Fig. 15 and show both the initial solution corresponding to the initial conditions, and that the output constituting the solution of the differential equation runs indefinitely without apparent losses or gains within the limitation of quantizing errors. In an actual electronic circuit realisation to give effect to the solution of the second order differential equation, the use of real capacitors with losses, for example, would result in the progressive decay or growth of the output signal. However, with the above described arrangement because the solution is being continually calculated, there is no apparent decay, nor any unstable growth leading to saturation. This represents a substantial improvement over the prior art analogue computers.

[0067] Fig. 16 is an example of a more complex array able to be compiled in accordance with the preferred embodiment of the present invention. It will be seen from Fig. 16 that four Fast Fourier Transforms, six different signal generators, numeric displays, bar charts, four reference voltages, four 40dB/decade filters, one "area under the curve" calculator and numerous comparators, multipliers, and the like are all provided.

5 [0068] Because the results, such as the waveforms illustrated in Fig. 6, are stored in memory, the user is able to replay these stored real time results. Furthermore, various portions of the results are able to be extracted for any specific purpose and even fed back into the apparatus using a "from disc" icon. This allows the signal processing to be adapted so that a desired feature of the input signal will be detected. Also the stored result can be continually replayed in real time, or faster or slower, as desired. Similarly, initial conditions can be established by using the previously stored result of a first array, as the initial input condition for a second array.

10 [0069] Furthermore, because the results are stored, the stored data is able to be edited and exported to spreadsheets, graphics or statistics utility programs such as EXCEL, LOTUS 123, and the like. This enables the graphical results of experiments to be combined with text describing the nature of the experiments and the nature of the conclusions.

15 [0070] The windows such as those indicated in Fig. 8, for example, are also compatible with multi media standards for the IBM PC. The specification for the window icon can be set up to be "PAL" which receives a conventional signal from a video camera, convert and then displays the video signal on the screen. If desired, this video signal can be frame grabbed to pixel video data which is able to be stored in the computer memory. If desired the conventional video camera signal can also be stored on a VCR at the same time. This stored video data can be replayed in the same manner as any other waveform captured by the apparatus. In this way, video picture correlation with other signal waveforms can be achieved. The time stamp placed on the conventional VCR video tape is, in this embodiment, synchronised with the signal data captured by the apparatus and stored therein. This arrangement is particularly advantageous since it enables the electrical results to be clearly identified with the optical record of the events which created the results.

20 [0071] The provision of icon specification means that the apparatus is particularly adaptive. For example, an array can be formed with a view to carrying out an analysis of, for example, the electrical voltages produced by the human heart. If as a result of the analysis of the results of these experiments, it is thought that a particular drug may produce some beneficial effect, then the array itself need not be changed in order to permit the same measurement to be carried out on a rat to which the drug has been administered. All that is required to be changed is the specification of those icons which are used, for example, as a reference. For example, the actual voltage level and the frequency can be adjusted in order to account for the different electrical outputs (eg voltage levels and different pulse rates) between humans and rats.

25 [0072] Furthermore, the ability to create arrays at will means that the apparatus is able to measure the association or independence between signals and so add a further dimension to the results able to be achieved. For example, a heart rate can be correlated with, or multiplied by, a respiration rate of a patient to give a third signal which can be regarded as a signature signal representing a result of significance to the user. Many such signals, even of mixed units, can be associated in this manner, if necessary scaled appropriately, and then compared against past data stored in memory.

30 [0073] It will be apparent to those skilled in the art, that the above described system has integrated all functions that are required for an analog workstation. Hitherto, there has not been a completely integrated system. Instead a plethora of partial functions existed.

35 [0074] In particular the following features are component parts of the integrated whole of the scientific emulator of the preferred embodiment.

40 1 A multiple purpose/multifunction analog module which inputs/outputs digital and analog signals and other functions.

45 2 A "real time" video, both in PAL or picture form at the same time and waveform displays in up to 40 windows on one screen are available. The oscilloscopic displays (one kind of display methodology) are able to display sample rates of the order of 25KHz or better.

50 3 Signal processing is not controlled by a prose style (von Neuman) sequentially syntactic "line language" but by a simply understood and readily used parallel graphic compiler which is able to be used by non programmers.

4 Waveforms and graphic arrays in pictorial form are readily transported to existing PC tools such as wordprocessors, spreadsheets and "offline" analysis software for report generation/records etc.

5 Non volatile storage can be accessed in such a way as to be able to "source" data that is to be fed back into an array for other results. Also correlating a waveform template, stored on disc, against any incoming waveform to achieve shape detection is available. (Template matching).

55 6 Network compatability. The system is able to have ist various functions distributed over a network. That is, storage can be performed at the PC in an office, or waveform data can be sourced on one PC to be displayed on another. Also the program and data can be transported over the network to another similar scientific instrument emulator

for analysis. The data and program are linked and are "bundled" for network transmission.

- [0075] In its preferred form, the system takes the form of a signal processing device comprising proprietary hardware and software which is contained in an IBM PC with data acquisition amplifiers positioned in floppy disk nacelles and processor PCBs and real time video cards fitted onto the PC system bus. The device in one configuration contains an extra 2 ordinary General purpose processors and a further 2 signal processors. These combined processors are programmed by the use of a graphical compiler on the VGA screen in such a way that the PC host (operating at say three million instructions per second) views the resident device subsystem (operating say at fifteen million instructions per second) as an extension of its memory. All the other processors also see each other as extensions of their own memory. In this way the multitasking software can manage multiple processors with each task on the various processors communicating with the other task(s) and/or processor(s) by passing pointers to shared memory. Thus no special communications hardware or software is required. The graphical compiler which compiles executable object code for the installed subsystem, allows the rigorous use of mathematics which is processed so fast as to simulate real time with all necessary processing performed during a single sample time.
- [0076] In addition to the functional adequacy of the programming system of the graphic compiler, the programming system using icons is constructed in such a way that compiled icons can themselves be composed of simple icons. This allows mathematical construction to be applied to two types of data streams involved with the processing. One of these types of data streams is the "acquired data stream" from the analogue to digital inter-conversion process and the other type is a "trigger stream". A trigger stream is some signal acquired from this inter-conversion and with which other signals which are desired to be displayed in time relationship, or is created by a timebase established in the array. Most often the trigger stream is created by a processing function wherein processing on one or many channels creates a signal which is intended to trigger displays, trigger processing such as averaging, or trigger storage. In general it is the complex trigger capability of the system, simply represented by the graphical compiler, which allows the accumulation of parameters which characterise various waveforms or epochs of waveforms. This allows their selection by experts to establish an "example set".

INDUSTRIAL APPLICATION

- [0077] The cost effectiveness of the apparatus of the preferred embodiment can be considered from the following. The apparatus can function as any one of the following devices, or any combination of the following devices (or multiples thereof) arranged together into an array.

	DEVICE	COST A\$
1	4 Channel Chart Recorder (500 Hz/channel)	10,000
2	FFT analyser (1Hz - 20KHz, max 512 points, 200 analyses sec.)	13,000
3	Frequency Counter (0.01% accuracy, 0.01 Hz - 10 MHz)	1,000
4	Function Generator (Sine, square, ramp, and triangle waveforms, 0.01Hz - 2MHz)	5,000
5	Data Logger (PC based)	4,000
6	Energy Monitor (Volts x Current, isolated inputs)	4,000
7	2 Channel Transient Recorder	3,500
8	Electrophysiological Monitor (ECG, EEG, EMG, ERP)	20,000

- [0078] The above items of equipment constitute a cost of approximately A\$60,500, however, the retail price of a scientific instrument emulator in accordance with the preferred embodiment and able to emulate the above functions is in the vicinity of A\$15,000 (assuming the user has an appropriate personal computer to both receive the necessary hardware and run the necessary software).

- [0079] Furthermore, the large number of parallel inputs in the integrated system is well suited to the study and analysis of parallel systems such as ecosystems, biosystems, machine systems, etc.

- [0080] The foregoing describes only one embodiment of the present invention and modifications, obvious to those skilled in the art, can be made thereto without departing from the scope of the present invention.

COPYRIGHT NOTICE

- [0081] The program listings contained in Appendices I-IV are the subject of copyright owned by the applicant and are not to be reproduced in any way without the express prior written authority of the applicant.

APPENDIX I

[0082]

Code for Sheets.

The following code is used to define a sheet in which the user specifies the parameters for a triggered pulse generator:-

```

5
10
15
20
25
30
35
40
45
50
55

/*****
 *          AMLAB - Associative Measurement Laboratory
 *
 *          PULSE Icon Overlay Sheet
 */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <dir.h>
#include "library.h"
#include "pulsegen.h"

/*****
 *          Window Dimension Constants
 *****/
#define WINDOW_WIDTH      55
#define WINDOW_HEIGHT     21
#define USER_X_SIZE      WINDOW_WIDTH
#define USER_Y_SIZE      (WINDOW_HEIGHT - 1)
/*
#####
####          PULSE Sheet Parameter Definition
#####
*/
typedef struct
{
    ICON_PARAM          ip;
    PULSE_ICON_PARAM    pa;
} PULSE_SHEET;

PULSE_SHEET pulse_vars =
{
    {ICON_PARAMETER_REC.
1, sizeof(PULSE_SHEET), {0}, sizeof(PULSE_ICON_PARAM)},
    {1000, MAX_STATE_LEVEL, MIN_STATE_LEVEL, 0,
    {POSITIVE_EDGE, 0}, 0, NORMAL_TRIGGER}
};

```

```

/*
#####
5  #####      PULSE Sheet WINDOW Definition
#####
*/
/*
10  #####      PULSE sheet Variables
#####
*/
/* Local string Arrays.      */
15 char  tilte[] = "PULSE GENERATOR SHEET";

/* ##### Sheet Title Definition #####      */
20 TITLE pulse_sheet_title =
{
    NULL,
    -1,
    BLACK,WHITE,
25  tilte,
    NULL
};

/* ##### Sheet Control Definition #####      */
30 CONTROL pulse_sheet_control =
{
    NULL,
    -1,
35  IWHITE,GREY,
    LOGO,
    0,0,
    0,NULL,
40  0,0,0,0,
    0,0,0,0
};

/* ##### Edit Region Structure Definitions. #####
*/
int  check_duration(int);
EDIT_DEFN pulse_duration =
50  {
    UNSIGNED_INTEGER,
    5,
    BLACK,WHITE,IWHITE,GREY,
55  0,0,

```

```

5      "Pulse ON State Duration (Samples): ",
      &pulse_vars.pa.duration,
      check_duration,
      NULL
      };

10     int    check_on_level(int);
      EDIT_DEFN pulse_on_level =
          {
          INTEGER,
15         6,
          BLACK, WHITE, IWHITE, GREY,
          0, 0,
          "Pulse State Levels (mV) - ON: ",
20         &pulse_vars.pa.on_level,
          check_on_level,
          NULL
          };

25     int    check_off_level(int);
      EDIT_DEFN pulse_off_level =
          {
          INTEGER,
30         6,
          BLACK, WHITE, IWHITE, GREY,
          0, 0,
          "- OFF: ",
35         &pulse_vars.pa.off_level,
          check_off_level,
          NULL
          };

40     int    check_trigger_level(int);
      EDIT_DEFN pulse_trigger_level =
          {
          INTEGER,
45         6,
          BLACK, WHITE, IWHITE, GREY,
          0, 0,
          "Trigger Level (mV): ",
50         &pulse_vars.pa.trigger_level,
          check_trigger_level,
          NULL
          };

55

```



```

5      int    check_edge(int);
      EDIT_DEFN pulse_trigger_edge =
          {
              CHAR,
              1,
10          BLACK, WHITE, IWHITE, GREY,
              0,0,
              "Edge (+,-): ",
              &pulse_vars.pa.edge,
              check_edge,
15          NULL
          };

      int    check_slew(int);
20      EDIT_DEFN pulse_trigger_slew =
          {
              UNSIGNED_INTEGER,
              5,
25          BLACK, WHITE, IWHITE, GREY,
              0,0,
              "Slew (mV/mS): ",
              &pulse_vars.pa.slew,
              check_slew,
30          NULL
          };

      int    check_mode(int);
35      EDIT_DEFN pulse_trigger_mode =
          {
              CHAR,
              MAX_TRIG_MODE,
40          BLACK, WHITE, IWHITE, GREY,
              0,0,
              "Mode (SINGLE SWEEP, NORMAL): ",
              &pulse_vars.pa.trigger_mode,
              check_mode,
45          NULL
          };

50      /* ##### Sheet Region Definitions. ##### */
      SCREEN_REGION pulse_sheet_regions[] =
          {
55          EDIT_CLASS,

```

```

5      4,2,0,0,
      &pulse_duration,

      EDIT_CLASS,
      4,5,0,0,
      &pulse_on_level,
10
      EDIT_CLASS,
      28,7,0,0,
      pulse_off_level,
15
      EDIT_CLASS,
      4,10,0,0,
      &pulse_trigger_level,
20
      EDIT_CLASS,
      12,12,0,0,
      &pulse_trigger_edge,
25
      EDIT_CLASS,
      12,14,0,0,
      &pulse_trigger_slew,
30
      EDIT_CLASS,
      4,17,0,0,
      &pulse_trigger_mode
35      };

#define      NR_REGIONS      (sizeof(pulse_sheet_regions) /
      sizeof(SCREEN_REGION))

40      /*      #####      PULSE Sheet Window Definition.      #####
      */
      AMWINDOW pulse_sheet_window[] =
45      {
      0,0,0,0,
      (VIRTUAL_MAX_X / 2), (VIRTUAL_MAX_Y / 2),
      WINDOW_WIDTH, WINDOW_HEIGHT,
      0,0,
50      BLACK, WHITE,
      NULL,
      NULL,
      NULL,
55      &pulse_sheet_title,

```

```

5      &pulse_sheet_control,
      0,NULL,
      NR_REGIONS,pulse_sheet_regions
      );

10  /* ##### Error Messages. ##### */
      ERROR_MSG      pulse_sheet_errors[] = {
          /* 0 */
          RANGE_ERROR,
          "Pulse Duration Error.\n\nThe Pulse \
15  Duration specified is Invalid.\n\nValid range is from 1 to 65535 samples",

          /* 1 */
          RANGE_ERROR,
          "ON State Level Error.\n\nThe ON state \
20  level specified for the pulse is Invalid.\n\nValid Range is -10000 to 10000
          mV.",

          /* 2 */
          RANGE_ERROR,
          "OFF State Level Error.\n\nThe OFF state \
25  level specified for the pulse is Invalid.\n\nValid Range is -10000 to 10000
          mV.",

          /* 3 */
          RANGE_ERROR,
          "Trigger Level Error.\n\nThe Trigger \
30  level specified is Invalid.\n\nValid Range is -10000 to 10000 mV.",

          /* 4 */
          RANGE_ERROR,
          "Edge Error.\n\nThe Trigger Edge \
40  specified is Invalid.\n\nValid Edges are + or -.",

          /* 5 */
          RANGE_ERROR,
          "Trigger Mode Error.\n\nThe Mode \
45  specified is Invalid.\n\nValid Modes are SINGLE SWEEP or NORMAL.",

          /* 6 */
          RANGE_ERROR,
          "Slew Rate Error.\n\nThe Slew \
50  rate specified is Invalid.\n\nValid Range is 0 to 10000 mV/mS."      };
55

```

```

/*
#####
5  #####      PULSE Sheet Definition
#####
*/
SHEET          pulse_sheet =
10              {
                pulse_sheet_window,
                &pulse_vars.ip,
                pulse_sheet_errors
15              };

/* Local Save Area of Global Variable Pointer. */
GLOBAL_VARS    *global_variables;
20

/*****
*
*          PULSE Sheet Handler
*
25 *****/
/*
SHEET          *pulse_sheet_handler(GLOBAL_VARS    *global)
{
30      /* preset the global variables static */
      global_variables = global;

      return( sheet_handler( global, USER_X_SIZE, USER_Y_SIZE,
35      NR_REGIONS, &pulse_sheet, pulse_sheet_regions ) );
}

/*****
*
*          Check Duration      (0)
40 *****/
/*
int    check_duration(int    checkpoint)
{
45      return( check_range( checkpoint, &pulse_vars.pa.duration,
                          MAX_UNSIGNED, 1, 0 ) );
}

50 etc...

```

55

APPENDIX II

[0083]

5

Compiler Elements

The following code is an example of the compiler element used for a triggered pulse generator:-

10

15

20

25

30

35

40

45

50

55

```

/******
 *                               Pulsegen Icon Compiler Overlay
 ******
 */
#include      <stdio.h>
#include      <stdlib.h>
#include      "compiler.h"
#include      "library.h"
#include      "comms86.h"
#include      "pulsegen.h"
/******
 *                               Local Definitions
 ******
 */
void  init_trigger_params(PULSE_ICON_PARAM *);

/*
#####
####          Pulse Process
#####
 */
#define      NR_PULSE_INPUTS          1
#define      NR_PULSE_OUTPUTS        1

/* I/O Buffer Lists.  */
unsigned  int  pulse_input_ids[NR_PULSE_INPUTS] = {1};
unsigned  int  pulse_output_ids[NR_PULSE_OUTPUTS] = {0};

/* Cell Parameters.  */
struct PULSE_struct  pulse_param;

/* Processes in this Icon.  */
struct proc_type      pulse_proc =
{
    NULL,              /* the next process defined */
    PULSE,             /* the cell required */
    pulse_input_ids,   /* array of input buffer numbers */
    pulse_output_ids,  /* array of output buffer numbers */
    &pulse_param,      /* array of the initial values of the cells parameters */

```

```

0          /* the base xdrum address of variables for this cell */
1;

5
/*
#####
####      Trigger Process
#####
10
*/
#define          NR_TRIGGER_INPUTS          1
#define          NR_TRIGGER_OUTPUTS        1

15
/* Trigger input ID specifications. */
unsigned int    trig_input_ids[NR_TRIGGER_INPUTS] = {0};

/* Trigger output ID specifications. */
20
unsigned int    trig_output_ids[NR_TRIGGER_OUTPUTS] = {1};

/* Trigger parameters. */
25
struct THRESH_struct    trig_params;

/* Process Descriptor. */
struct proc_type trigger_proc =
30
{
    &pulse_proc,          /* the next process defined */
    THRESH,              /* the cell id required */
    trig_input_ids,      /* array of input buffer numbers */
    trig_output_ids,     /* array of output buffer numbers */
    &trig_params,        /* array of the initial values of the cells
35
                        parameters */
    0                    /* the base xdrum address of variables for this
                        cell */
40
};

/*
#####
####      PULSEGEN CONFIGURATION
#####
45
*/
unsigned int    *int_interconnect_ids[] =
50
{
    &trig_output_ids[0],
    &pulse_input_ids[0]
};

55

```

```

5      #define      NR_INTERCONNECT_IDS (sizeof(int_interconnect_ids) /
                                     sizeof(unsigned int *))

      unsigned      int      *ext_input_ids[] =
      {
10          &trig_input_ids[0]
      };

      #define      NR_EXT_INPUT_IDS      (sizeof(ext_input_ids) /
                                     sizeof(unsigned int *))

15      unsigned      int      *ext_output_ids[] =
      {
          &pulse_output_ids[0]
20      };

      #define      NR_EXT_OUTPUT_IDS      (sizeof(ext_output_ids) /
                                     sizeof(unsigned int *))

25      /*      PULSEGEN Dsp System Configuration.      */
      ICON_CONFIG      pulse_config =
      {
30          2,          /*      nr_processes      */
          &trigger_proc, /*      process list      */
          0,          /*      nr_raw      */
          NULL,        /*      raw list      */
          0,          /*      nr_screen      */
35          NULL,        /*      screen list      */
          0,          /*      nr_virtual      */
          NULL,        /*      virtual list      */
          NR_INTERCONNECT_IDS, /* number of int interconnect ID's. */
          int_interconnect_ids,
40          NR_EXT_INPUT_IDS, /* number of external input ID's. */
          ext_input_ids,
          NR_EXT_OUTPUT_IDS, /* number of external output ID's. */
          ext_output_ids,
45          NULL,          /* Overlay Struct      */
          NULL,          /* Icon parameter key */
          NULL,          /* next_proc      */
          NULL,          /* next_raw      */
50          NULL,          /* next_scrn      */
          NULL,          /* next_virt      */
          NULL };        /* next_icon      */
55

```

```

5  /******
   *
   * Request Block Size
   *
   */
   unsigned    int    mem_block_size = 0;

10 /******
   *
   * Pulsegen Descriptor Handler
   *
   */
15 void    *pulsegen(GLOBAL_VARS *gv, void *mem_block, ICON_PARAM
   *ip)
   {
20     PULSE_ICON_PARAM    *pip;

   /* Check for memory block request. */
   if (mem_block == NULL)
       return(&mem_block_size);

25     /* Index pulse icon parameters. */
     pip = (PULSE_ICON_PARAM *) &tip[1];

30     /* Set the pulse state levels. */
     pulse_param.WIDTPU = pip->duration;
     pulse_param.HIGHPU = calc_AD_val(pip->on_level);
     pulse_param.LOW_PU = calc_AD_val(pip->off_level);

35     /* Set the Trigger Cell Variables. */
     init_trigger_params(pip);

40     /* Return Configuration Pointer. */
     return(&pulse_config);
   }

45 /******
   *
   * Init Trigger Params
   *
   */
   void    init_trigger_params(PULSE_ICON_PARAM *pip)
   {
50     /* Transfer the level parameter. */
     trig_params.THRETH = calc_AD_val(pip->trigger_level);

55     /* Preset trigger width. */

```



```

trig_params.WINSTH = 1;

5
/* Transfer the Slew Parameter. */
trig_params.SLEWTH = pip->slew;

10
/* Preset Dead Period. */
trig_params.DEADTH = 0;

/* Transfer the Trigger Edge Setting. */
15
if (pip->edge[0] == POSITIVE_EDGE)
    trig_params.POSTTH = FLAG_POS_EDGE;
else
    trig_params.POSTTH = FLAG_NEG_EDGE;

20
/* Transfer the One Shot Status. */
if (strcmp(pip->trigger_mode, SINGLE_SWEEP_TRIGGER) == 0)
    trig_params.ONESTH = FLAG_ONE_SHOT;
else
25
    trig_params.ONESTH = FLAG_CONTINUOUS;
}

30

35

40

45

50

55

```

APPENDIX III

[0084]

5

Cell Code

10

The following code is used to implement the Pulse Generator on the DSP chip. It is written in TMS320 assembler code.

15

```

*
*      Cell  :-  PULSE
*****
*
*      Synopsis
* This cell generates a pulse once every time a trigger occurs.
* It has one input which is a trigger stream.
* It has one output which is the pulse train produced.
*
* The cell has the following functioning :-
*
*      read in the trigger buffer
*
*      for each point in the buffer:-
*
*          if waiting for the trigger to go high:-
*
*              if trigger is high:-
*
*                  change state to 1.
*                  reset pulse width counter.
*
*              else if trigger is low
*
*                  change to state 0
*
*              if( count > 0 )
*
*                  count--;
*                  output = high
*
*              else output = low.
*
*      finish the cell
*****
*
IDT  PULSE
COPY WHERELIB

```

55

5 *

 *

 SYSTEM VARIABLES

 *

10 REF RG1 WORKING REGISTERS

 REF RG2

 REF RG3

15 REF ONE THE NUMBER ONE

 REF BUFSIZ THE SIZE OF BUFFERS

 REF I A COMMONLY USED LOOP COUNTER

 REF FREESH THE ALLOC_BASE_PTR

20 REF ZERO ZERO

 *

 SYSTEM CALLS

25 *

 REF ENDCEL THE END OF THIS CELL

 *

30 *

 *

 LIBRARY ROUTINES

 *

35 REF READCH READ IN A CHANNEL

 *

40 *

 *

 THIS CELLS VARIABLES

 *

45 REF TRIGPU INPUT. THE CELLS TRIGGER

 *

 REF OUTSPU OUTPUT. THE CELLS OUTPUT.

50 REF BUFFPU TEMP ARRAY. USED TO BUFFER

 *

 REF STATPU POINTS.

 *

 REF STATPU STATIC. STATE OF TRIGGER

 *

 REF WIDTPU DETECTION.

55 REF WIDTPU PARAM. WIDTH OF PULSE.

```

5      REF  HIGHPU      PARAM. HIGH PULSE LEVEL.
      REF  LOWSPU       PARAM. LOW PULSE LEVEL.
      REF  COUNPU       STATIC. THE NUMBER OF POINTS
                          LEFT IN THE PULSE
      *
      *
      *****
10     *
      *   CONSTANTS
      *
      *****
15     *
      *   COPY CASPDEF
      *
      *****
20     *
      *   PULSE.
      *       INPUTS
      *           TRIGPU
25     *
      *       OUTPUTS
      *           OUTSPU
      *
      *****
30     *
      *   DEF PULSE
      *
      *   PULSE
35     *
      *       read in the trigger buffer
      *
      *       LARK AR1,BUFFPU
      *       LAC TRIGPU
40     *       SACL RG1
      *       CALL READCH
      *
      *       for each point in the buffer:-
45     *   for( i = 0 ; i < buf_size ; i++ )
      *
      *       LAR AR0,BUFSIZ
      *       LARK AR1,BUFFPU
50     *       LAC OUTSPU
      *       ADD ONE,1      LINE UP WITH DATA
      *       SACL RG1
      *       SXRAM      RG1
55

```

```

*
PULSF1    BANZ PULSC1
*
5          B    ENDCEL
PULSC1
*
10         *    if trigger is high:-
*
*
* READ IN THE TRIGGER
*
15         LARP AR1
          LAC  *+,0,AR0
          BZ   ELSE1
*
20         *
*          if waiting for the trigger to go high:-
*
          LAC  STATPU
          BNZ  DOOUT
*
25         *
*          change state to 1.
*          reset pulse width counter.
*
30         LAC  ONE
          SACL STATPU
          LAC  WIDTPU
          SACL COUNPU
          B    DOOUT
*
35         *
*          else change to state 0
*
40         ELSE1ZAC
          SACL STATPU
*
*          if( count > 0 )
*
45         DOOUT    LAC  COUNPU
          BLEZ ELSE2
*
*
50         *          count--;
*          output = high
*
          SUB  ONE
          SACL COUNPU
55

```

EP 0 573 503 B1

```
5          WXDRAM  HIGHPU
          B        PULSF1
          *
          *
          *      else output = low.
          *
10          NOP
          ELSE2 WXDRAM  LOWSPU
          B        PULSF1
          *
          *      finish the cell
          *
15          END
```

20

25

30

35

40

45

50

55

APPENDIX IV

[0085]

5 **DSP System Code**

10 The following code implements the system used by the DSP's in an AMLAB environment:-

```

*
*       MODULE:-  SYSTEM
*
15 *****
*
*                       SYNOPSIS
*
20 * DUE TO THE PERFORMANCE REQUIREMENTS OF THE TMS320 IT
* IS NECESSARY TO IMPLEMENT EACH PROCESSING ELEMENT AS
* A CELL IN A INTERPRETER TYPE SITUATION. AN EXPLANATION
* OF THIS INTERPRETER FOLLOWS. NO ATTEMPT TO EXPLAIN
25 * WHY THINGS ARE DONE THE WAY THEY ARE IS GIVEN HERE AS
* THESE CAN BE FOUND IN THE ACCOMPANYING
* SPECIFICATIONS.
*
* DEFINITION OF TERMS
30 *
* CELL                A PROCESSING ELEMENT. THE TOTAL
*                       PROCESSING TO BE DONE IS
*                       BROKEN DOWN INTO MODULES THAT ARE
35 *                       EASY TO IMPLEMENT.
*
* EVENT               AN EVENT IS SOMETHING THAT HAPPENS. IT
*                       MAYBE INTERNAL TO THE SYSTEM SUCH AS A
*                       BEAT IS PROCESSED TO A CERTAIN LEVEL OR
40 *                       EXTERNAL SUCH AS AN INTERRUPT. THIS
*                       EXECUTIVE IS AN EVENT DRIVEN ONE. IN
*                       THAT IT IS EVENTS WHICH ULTIMATELY
*                       DETERMINE WHICH SEQUENCES ARE RUN AND
45 *                       IN WHAT ORDER.
*
* SEQUENCE            A PRE-DETERMINED ORDER IN WHICH CELLS
*                       ARE RUN.
50 *
* ACTIVE              THE ACTIVE CELL IS THE CELL CURRENTLY
*                       BEING EXECUTED OR ABOUT TO BE
*                       EXECUTED
55 *
* CCB                 A CELL CONTROL BLOCK. THIS CONTAINS ALL THE

```

* RELEVANT INFORMATION ABOUT A CELL NEEDED
 * BY THE SYSTEM.
 *
 5 * ENDSCEL THE ROUTINE WHICH TIDIES UP AFTER A CELL
 * RELINQUISHES CONTROL, FINDS THE NEXT CELL IN
 * THE SEQUENCE, AND INSTALLS THAT CELL.
 *
 10 * DESCRIPTION OF THE SYSTEM
 *
 * THE SYSTEM COULD IN BROAD TERMS BE DESCRIBED AS AN
 * INTERPRETER. CELL 0 IS THE HIGHEST LEVEL OF PROCESSING
 15 * IN THE SYSTEM. IT DETERMINES WHICH SEQUENCES
 * WILL BE RUN BASED ON THE INFORMATION CONTAINED IN THE
 * STATUS. SEQUENCES IN TURN ARE COMPOSED OF CELLS. CELL
 * 0 RUNS A SEQUENCE. THE SYSTEM TAKES THAT SEQUENCE
 20 * AND RUNS EACH CELL IN TURN. INDIVIDUAL CELLS CAN
 * CHANGE THE STATUS TO INDICATE TO CELL 0 WHAT IS THE
 * RESULTS OF ITS PROCESSING. FURTHER MORE CELLS ARE ABLE
 * TO ABORT THE CURRENT SEQUENCE WHICH RETURNS
 25 * CONTROL TO THE CELL 0. CELL 0 IS DIFFERENT TO THE OTHER
 * CELLS IN THAT IT ALWAYS SAVES A RETURN ADDRESS BEFORE
 * RELINQUISHING CONTROL. THIS IS BECAUSE CELL 0 IS NEVER
 * COMPLETED, RELINQUISHING CONTROL FREQUENTLY, AND
 30 * RESTARTING WHERE IT LEFT OFF THIS ALLOWS US TO
 * IMPLEMENT A MACRO LEVEL PROGRAM IN CELL 0, COMPOSED
 * OF SEQUENCES WHICH IN TURN ARE COMPOSED OF CELLS.
 *
 35 * SEQUENCES ARE REFERENCED BY THE SEQUENCE REFERENCE
 * LIST, WHICH, FOR EASE OF PROGRAMMING EXISTS AT THE
 * BOTTOM OF EXTERNAL DATA RAM. EACH MEMBER IN
 * THIS LIST POINTS TO A SEQUENCE LIST. EACH SEQUENCE LIST
 40 * MEMBER IS A POINTER TO CELL CONTROL BLOCK. EVERY
 * SEQUENCE HAS A SEQUENCE LIST. EACH CONSECUTIVE
 * CELL IN A SEQUENCE IS POINTED TO BY A CONSECUTIVE
 * MEMBER IN THE SEQUENCE LIST
 *
 45 * A CELL IS RESTORED BY LOADING INTERNAL DATA MEMORY
 * WITH THE CELLS STATIC VARIABLES. THESE ARE POINTED TO
 * BY A MEMBER OF THE CCB. THE CELLS START ADDRESS IS
 * ACCESSED BY USING THE CELLID, ANOTHER MEMBER OF THE
 50 * CCB, AS AN OFFSET TO A LOOK UP TABLE STORED IN PROGRAM
 * MEMORY CALLED THE CSASB. A CELL IS SAVED BY THE
 * REVERSE PROCESS. THE FINAL ELEMENT OF A CCB IS A
 55 * CONSTANT. A CONSTANT IS A STATIC VARIABLE THAT DOES

* NOT CHANGE ITS VALUE AND HENCE DOES NOT NEED TO BE
* STORED AWAY.

IDT 'SYSTEM'
COPY WHERELIB

STATIC VARIABLES USED BY THE SYSTEM

REF	CCBSPT	A POINTER TO THE CURRENTLY ACTIVE CCB
REF	CURSSQ	A POINTER TO THE CURRENTLY ACTIVE SEQUENCE
REF	ONE	
REF	TABLE	A POINTER TO THE VARIABLES TABLE IN IORAM
REF	RG1, RG2	
REF	CSASB	THE CELL ADDRESS BLOCK
DEF	ABORT	LET OUTSIDE TASKS KNOW ABOUT THE ROUTINES
DEF	ABORT1	
DEF	ENDCEL	
DEF	RUN	

CONSTANTS USED BY THE SYSTEM

REF	CELVAR	THE START OF A CELLS VARIABLES IN INTERNAL RAM
REF	C0CCB	THE ADDRESS OF CELLO CCB.

THE SYSTEM ENTRY POINT. THIS IS ONLY USED ON START

* UP.

```

*
*****
*
5      DEF SYSTEM
      PSEG
SYSTEM  B    ABORT!    SYSTEM ENTRY POINT
*
10     *****
*
*      SAVE THE ENVIROMENT OF THE CURRENT CELL
*
15     *****
*
SAVE RXDRAM  RG2,CCBSPT    READ IN THE CELL ID
      RXDRAM  TABLE      READ IN THE CURRENT
20     *                               CELLS VARIABLE TABLE.
      RXDRAM  RG1          READ IN THE NUMBER OF
*                               INSTALLED VARIABLES
      RXDRAM  RG1          READ IN THE NUMBER OF
25     *                               STORED VARIABLES
*
      LAR  AR0,RG1
*
30     LAR  AR1,CELVAR      SET UP THE POINTER TO
*                               THE VARIABLES
*
      SXRAM  TABLE
35     LARP 0
      BANZ SAVELP          ARP = 0
*                               IF THERE ARE ANY MORE
      B      SAVRET        VARIABLES SAVE THEM
40     *
SAVELP
      LARP 1
      WXDRAM  *+,AR0      WRITE OUT THE NEXT
45     *                               VARIABLE
      BANZ SAVELP          ARE THERE ANY MORE
*
SAVRET
50     RET
*
*****
*
55     *      RESTORE THE ENVIROMENT OF THE CURRENT CELL

```

```

*
*****
5  *
  RESTOR
      RXDRAM  RG2,CCBSPT  READ IN THE CELL ID
      RXDRAM  TABLE      READ IN THE CURRENT
10  *                      CELLS VARIABLE TABLE
  *                      ADDR.
      RXDRAM  RG1          READ IN THE NUMBER OF
  *                      VARIABLES
15  *
      LAR  ARO,RG1
  *
      LAR  ARI,CELVAR      SET UP THE POINTER TO
20  *                      THE VARIABLES
      SXRAM  TABLE
      LAR 0
      BANZ RESTLP          ARP = 0
25  *                      IF THERE ARE ANY MORE
      B  RESRET            VARIABLES SAVE THEM
  *
  RESTLP
30  *      LAR 1
      RXDRAM  *+,ARO      WRITE OUT THE NEXT
  *                      VARIABLE
      BANZ RESTLP          ARE THERE ANY MORE?
35  *
  RESRET
      LACKB  CSASB        FIND THE RETURN
  *                      ADDRESS FROM THE CSASB
      ADD  RG2            THE CELLS ID
40  *      TBLR  RG1        READ IN THE RETURN
  *                      ADDRESS
      LAC  RG1
      PUSH
45  *      RET              PUSH IT ONTO THE STACK
                          AND GO THERE
50
55

```

```

*****
*
5  * ABORT THE CURRENTLY ACTIVE SEQUENCE AND RETURN TO
  * CELL 0 THIS ROUTINE SHOULD BE BRANCHED TO.
  *
*****
10 *
   ABORT
       CALL SAVE           SAVE THE ENVIROMENT OF THE
                           CURRENTLY ACTIVE CELL.
15 * ABORT1
       LACB      COCCB      SET THE CCB POINTER TO CELLO
                           CCB
   *
       SACL CCB$PT
20 *
       B      RESTOR      AND RESTORE CELL 0
   *
*****
25 *
   *   END THIS CELL AND INSTALL THE NEXT ONE IN THE
   * SEQUENCE THIS ROUTINE SHOULD BE BRANCHED TO.
   *
*****
30 *
   ENDCEL
       CALL SAVE           SAVE THE ENVIROMENT OF THE .
                           LAST CELL
35 * FIND THE ADDRESS OF THE NEXT CELL IN THE SEQ.
   *
       RXDRAM      CCB$PT,CUR$SQ
   *
40 *   LAC  CUR$SQ      INCREMENT THE SEQ POINTER
       ADD  ONE
       SACL CUR$SQ
   *
45 *   B      RESTOR      INSTALL THE NEXT CELL
   *
50
55

```

```

*****
*
5  *   THIS ROUTINE IS ONLY EVER CALLED BY CELL 0. IT
*   STARTS OFF A NEW SEQUENCE.
*
*****
10 *
*   RUN  RXDRAM  CURSSQ.RG1RG1 CONTAINS THE SEQUENCE
*                                     NUMBER WHICH IS A
*                                     DIRECT OFFSET TO THE
15 *                                     SEQUENCE POINTER.
*                                     GET THE RETURN ADDRESS
*
*   POP
*   SACL RG2
*   LACKB  CSASB  AND WRITE IT OUT TO THE FIRST
20 *                                     LOCATION IN THE
*   TBLW RG2  CSASB
*
*   B  ENDCEL  AND PERFORM ENDCEL WHICH
25 *                                     WILL SAVE THE ENVIROMENT OF
*   CELL 0 AND THEN INSTALL THE FIRST CELL IN THE SEQUENCE
*
*****
30 *
*   PEND
*

```

35 Claims

1. A scientific instrument emulator comprising:
 - 40 a computer (1) including a memory (3) coupled to a processor (2);
 - a multifunctional input/output signal module (6) having at least one analogue signal input (7) with digitizing means (37) connected thereto and at least one signal output (8) and being configurable with respect to at least a sampling frequency of said analogue signal input (7);
 - 45 a video display (10) coupled to a video display generator (50), said video display generator (50) being coupled to said processor (2);
 - a library program stored in said memory (3), said library program including a plurality of instrumentation units each being able to be displayed on said video display (10) and each having a predetermined signal processing function;
 - 50 a set-up program stored in said memory (3) and carried out by said processor (2) to allow an operator to interconnect selected instrumentation units to form an array of interconnected instrumentation units, wherein a sampling frequency for said multifunctional input/output signal module (6) is specified, said set-up program enables signal feedback interconnection in said array, said array having an output being provided to at least one of the following: said video display, said multifunctional input/output signal module (6) and said memory (3); and
 - 55 a graphic compiler program stored in said memory (3) and executable by said processor (2) on completion of said set-up program to create a scientific instrument emulation program to carry out an overall signal processing function of said array of selected interconnected instrumentation units, said graphic compiler program being capable of resolving signal feedback in said array and to form a sequence in said emulation program

of said predetermined signal processing functions of said instrument units, which are each to be carried out once between successive samples of an input signal obtained using said multifunctional input/output signal module, said emulation program containing operations configured in accordance with said specified sampling frequency for sampling said input signal to provide a data stream from said multifunctional input/output signal module (6);

an instrument emulation sub-assembly coupled to said processor (2), said memory (3) and said video display generator, wherein said instrument emulation sub-assembly is directly connected to said multifunctional input/output signal module, said instrument emulation sub-assembly having:

a second processor for executing said scientific instrument emulation program to continuously carry out said overall signal processing function so that each sample of said data stream of successive samples is processed during a single sample time to produce an array output of said scientific instrument emulator before a successive sample of said input signal is obtained by said multifunctional input/output signal module (6);

wherein at least one of a plurality of operations is performed, said plurality of operations including each of displaying said array output on said video display (10) in real time, storing said array output in the memory (3), and providing said array output to said signal output of said multifunctional input/output signal module (6) in real time.

2. The scientific instrument emulator as claimed in claim 1, wherein said instrumentation units are each represented by a corresponding icon which is able to be displayed on said video display during operation of said set-up program whereby each desired instrumentation unit can be selected, located in, and interconnected with other instrumentation units in said array.

3. The scientific instrument emulator as claimed in claim 2, wherein said array includes a data signal feedback loop connected between an output of one of said instrumentation units and an input of one said instrumentation units.

4. The scientific instrument emulator as claimed in claim 1, wherein said multifunctional input/output signal module includes an additional signal output for connection to external hardware and that is available as a real time electric signal able to operate, or trigger, said external hardware.

5. The scientific instrument emulator as claimed in claim 4, wherein said multifunctional input/output signal module has both analogue and digital signal outputs.

6. The scientific instrument emulator as claimed in claim 4, wherein said array output stored in that memory is available for subsequent graphical manipulation and/or tabulation by said computer.

7. The scientific instrument emulator as claimed in claim 1, wherein said instrument emulation processor assembly further comprises a video generator connected to said second processor for providing a displayable representation of said array output before a succeeding sample is obtained.

8. A method for determining the execution order of predefined signal processing elements in a computer having a central processing unit (2) and electronic memory (3) to implement in real time a mathematical/signal processing function of an emulated instrument having at least one repetitively sampled signal input to form a signal output thereof, said method comprising the steps of:

applying a set-up program stored in said memory (3) and carried out by said central processing unit (2) to allow an operator to interconnect selected instrumentation units, wherein a sampling frequency for a multifunctional input/output signal module (6) is specified, said set-up program enables signal feedback interconnection in said array, said array having an output being provided to at least one of the following: a video display (10), said multifunctional input/output signal module (6) and said memory (3), each of said instrumentation units having a mathematical/signal processing function;

creating a scientific instrument emulation program using a graphic compiler program stored in said memory (3) and executable by said central processing unit (2) on completion of said set-up program, said scientific instrument emulation program carrying out said overall signal processing function of said array of selected interconnected instrumentation units;

determining an execution order of said mathematical/signal processing function including the steps of:

representing said function as a sequence of elemental mathematical steps in turn representable using predefined signal processing elements, and arranging the pre-defined signal processing elements in said sequence for sequential execution within the time between successive samples and

commencing with said at least one sampled signal input,

wherein said determining step thereby forms an array of instrumentation units having said overall circuit mathematical/signal processing function; carrying out said overall signal processing function so that each sample of said contiguous data stream is processed during a single sample time to produce an array output of said overall mathematical signal processing function before a successive sample of said input signal is obtained by said multifunctional input/output signal module (6);

representing said overall circuit mathematical/signal processing function as a sequence of events representable as predefined signal processing element events;

arranging said predefined signal processing element events in said sequence for sequential execution commencing with said at least one sampled array signal input, whereby a determinate computation time of the predefined signal processing elements is ensured by executing every one of said predefined signal processing elements once between successive samples, whereby said method is capable of resolving data signal feedback in said array of instrumentation units;

displaying said array output on said video display (10) in real time;

storing said array output in the memory (3); and

providing said array output to said signal output of said multifunctional input/output signal module in real time.

9. The method as claimed in claim 8, wherein a parameter of said functional block is able to be specified.

10. The method as claimed in claim 8, wherein said array includes at least one data signal feedback loop in which an output of one of said instrumentation units is connected to form an input of one of said plurality of instrumentation units, and an initial result of corresponding one of said executable events in said sequence of executable events is utilized in a repeated execution of an earlier event to generate a modified result of said corresponding one of said executable events from a previous sample period.

Patentansprüche

1. Wissenschaftlicher Instrumentenemulator, aufweisend:

einen Computer (1) mit einem Speicher (3), der mit einem Prozessor (2) gekoppelt ist;

ein multifunktionales Eingangs-/Ausgangs-Signalmodul (6), welches zumindest einen analogen Signaleingang (7) mit Digitalisierungsmitteln (37) hierzu verbunden und zumindest einen Signalausgang (8) aufweist und konfigurierbar ist mit Bezug auf zumindest eine Abtastfrequenz des analogen Signaleingangs (7);

eine Videoanzeige (10), die mit einem Videoanzeigengenerator (50) gekoppelt ist, wobei der Videoanzeigengenerator (50) mit dem Prozessor (2) gekoppelt ist;

ein Bibliotheksprogramm, das in dem Speicher (3) gespeichert ist, wobei das Bibliotheksprogramm eine Mehrzahl von Instrumentationseinheiten aufweist, wobei jede geeignet ist, auf der Videoanzeige (10) angezeigt zu werden, und jede eine vorbestimmte Signalverarbeitungsfunktion aufweist, ein Set-Up-Programm, das in dem Speicher (3) gespeichert ist, und durch den Prozessor (2) ausgeführt wird, um einem Bediener zu erlauben, ausgewählte Instrumentationseinheiten miteinander zu verbinden, um ein Feld von miteinander verbundenen Instrumentationseinheiten zu bilden, wobei eine Abtastfrequenz für das multifunktionale Eingangs-/Ausgangs-Signalmodul (6) spezifiziert ist, wobei das Set-Up-Programm Signale Rückführungsverbindungen in dem Feld ermöglicht, wobei das Feld einen Ausgang aufweist, der zumindest einem der folgenden zugeführt wird: der Videoanzeige, dem multifunktionalen Eingangs-/Ausgangssignalmodul (6) und dem Speicher (3); und

ein graphisches Compiler-Programm, das in dem Speicher (3) gespeichert ist und durch den Prozessor (2) ausführbar ist bei Vollendung des Set-Up-Programms, um ein wissenschaftliches Instrumentationsemulationsprogramm zu kreieren, um eine Gesamt-Signalverarbeitungsfunktion des Feldes von ausgewählten verbundenen Instrumentationseinheiten auszuführen, wobei das graphische Compiler-Programm in der Lage ist, Signale Rückkopplung in dem Feld aufzulösen und eine Sequenz zu bilden in dem Emulationsprogramm der vorbestimmten Signalverarbeitungsfunktionen der Instrumentationseinheiten, die jede auszuführen ist einmal zwischen sukzessiven Samples eines Eingangssignals, das erhalten wird unter Verwendung des multifunktionalen Eingangs-/Ausgangssignalmoduls, wobei das Emulationsprogramm Operationen aufweist, die ge-

mäß der spezifizierten Abtastfrequenz konfiguriert sind, zum Abtasten des Eingangssignals, um einen Datenstrom von dem multifunktionalen Eingangs-/Ausgangssignalmodul (6) zu schaffen;
eine Instrumentenemulationsunteranordnung, die gekoppelt ist mit dem Prozessor (2), dem Speicher (3) und dem Videoanzeigengenerator, wobei die Instrumentenemulationsunteranordnung direkt verbunden ist mit dem multifunktionalen Eingangs-/Ausgangs-Signalmodul, wobei die Instrumentenemulationsunteranordnung aufweist:

einen zweiten Prozessor zum Ausführen des wissenschaftlichen Instrumentenemulationsprogramms, um kontinuierlich die Gesamt-Signalverarbeitungsfunktion auszuführen, derart, dass jedes Sample des Datenstroms von sukzessiven Samples verarbeitet wird während einer einzelnen Sample-Zeit, um eine Feldausgabe des wissenschaftlichen Instrumentenemulators zu produzieren, bevor ein sukzessives Sample des Eingangssignals erhalten wird durch das multifunktionale Eingangs-/Ausgangssignalmodul (6);

wobei zumindest eine einer Mehrzahl von Operationen durchgeführt wird, wobei die Mehrzahl von Operationen jede von Anzeigen der Feldausgabe auf der Videoanzeige (10) in Echtzeit, Speichern der Feldausgabe in dem Speicher (3) und Bereitstellen der Feldausgabe zu dem Signalausgang des multifunktionalen Eingangs-/Ausgangssignalmoduls (6) in Echtzeit umfasst.

2. Wissenschaftlicher Instrumentenemulator gemäß Anspruch 1, bei welchem die Instrumentationseinheiten jede repräsentiert sind durch ein entsprechendes Icon, welches geeignet ist, auf der Videoanzeige angezeigt zu werden während des Betriebs des Set-Up-Programms, wobei jede gewünschte Instrumentationseinheit ausgewählt werden kann, die angeordnet ist in und verbunden ist mit anderen Instrumentationseinheiten in dem Feld.

3. Wissenschaftlicher Instrumentenemulator gemäß Anspruch 2, bei welchem das Feld eine Datensignal-Rückkopplungsschleife aufweist, die zwischen einem Ausgang einer der Instrumentationseinheiten und einem Eingang einer der Instrumentationseinheiten verbunden ist.

4. Wissenschaftlicher Instrumentenemulator gemäß Anspruch 1, bei welchem das multifunktionale Eingangs-/Ausgangssignalmodul einen zusätzlichen Signalausgang aufweist zur Verbindung mit externer Hardware und welcher verfügbar ist als ein Echtzeit-elektrisches Signal, welches geeignet ist, die externe Hardware zu bedienen oder zu triggern.

5. Wissenschaftlicher Instrumentenemulator gemäß Anspruch 4, bei welchem das multifunktionale Eingangs-/Ausgangssignalmodul sowohl analoge als auch digitale Signalausgänge aufweist.

6. Wissenschaftlicher Instrumentenemulator gemäß Anspruch 4, bei welchem die Feldausgabe, die in dem Speicher gespeichert wird, verfügbar ist für nachfolgende graphische Manipulation und/oder Tabulation durch den Computer.

7. Wissenschaftlicher Instrumentenemulator gemäß Anspruch 1, bei welchem die Instrumentenemulations-Prozessoranordnung des Weiteren einen Videogenerator aufweist, der mit dem zweiten Prozessor verbunden ist, zum Bereitstellen einer anzeigbaren Repräsentation der Feldausgabe, bevor ein nachfolgendes Sample erhalten wird.

8. Verfahren zum Bestimmen der Ausführungsreihenfolge vorbestimmter Signalverarbeitungselemente in einem Computer, der eine Zentralverarbeitungseinheit (2) und einen elektronischen Speicher (3) aufweist, um in Echtzeit eine mathematische/Signalverarbeitungsfunktion eines emulierten Instruments zu implementieren, welches zumindest eine wiederholt abgetastete Signaleingabe aufweist, um davon eine Signalausgabe zu bilden, wobei das Verfahren die folgenden Schritte aufweist:

Anwenden eines Set-Up-Programms, welches in dem Speicher (3) gespeichert wird und durch die Zentralverarbeitungseinheit (2) ausgerührt wird, um einem Bediener zu erlauben, ausgewählte Instrumentationseinheiten miteinander zu verbinden, wobei eine Abtastfrequenz für ein multifunktionales Eingangs-/Ausgangssignalmodul (6) spezifiziert wird, wobei das Set-Up-Programm Rückkopplungsverbindung in dem Feld ermöglicht, wobei das Feld einen Ausgang aufweist, welcher zumindest einem der folgenden zugeführt wird: einer Videoanzeige (10), dem multifunktionalen Eingangs-/Ausgangssignalmodul (6) und dem Speicher (3), wobei jede der Instrumentationseinheiten eine mathematische/Signalverarbeitungsfunktion aufweist;
Kreieren eines wissenschaftlichen Instrumentenemulationsprogramms unter Verwendung eines Compiler-Programms, welches in dem Speicher (3) gespeichert wird und durch die Zentralverarbeitungseinheit (2) bei Vollendung des Set-Up-Programms ausführbar ist, wobei das wissenschaftliche Instrumentenemulationspro-

gramm die Gesamt-Signalverarbeitungsfunktion des Feldes ausgewählter verbundener Instrumentationseinheiten ausführt,
Bestimmen einer Ausführungsreihenfolge der mathematischen/Signalverarbeitungsfunktion mit den folgenden Schritten:

5

10

15

20

25

Repräsentieren der Funktion als eine Sequenz elementarer mathematischer Schritte, die ihrerseits repräsentierbar sind unter Verwendung vordefinierter Signalverarbeitungselemente, und
Anordnen der vorbestimmten Signalverarbeitungselemente in der Sequenz für sequentielle Ausführung innerhalb der Zeit zwischen sukzessiven Samples und Beginnen mit der zumindest einen abgetasteten Signaleingabe,
wobei der Bestimmungsschritt hierdurch ein Feld von Instrumentationseinheiten bildet, welches die Gesamtschaltungsmathematische/Signalverarbeitungsfunktion aufweist;
Ausführen der Gesamt-Signalverarbeitungsfunktion, derart, dass jedes Sample des zusammenhängenden Datenstroms verarbeitet wird während einer einzelnen Abtastzeit, um eine Feldausgabe der gesamtmathematischen Signalverarbeitungsfunktion zu produzieren, bevor ein sukzessives Sample des Eingangssignals erhalten wird durch das multifunktionale Eingangs-/Ausgangssignalmodul (6);
Repräsentieren der Gesamtschaltungsmathematischen/Signalverarbeitungsfunktion als eine Sequenz von Ereignissen, die darstellbar sind als vorbestimmte Signalverarbeitungselementereignisse;
Anordnen der vorbestimmten Signalverarbeitungselementereignisse in der Sequenz für sequentielle Ausführung, beginnend mit der zumindest einen abgetasteten Feldsignaleingabe, wobei eine bestimmte Rechnerzeit der vorbestimmten Signalverarbeitungselemente gesichert wird durch Ausführen jedes der vorbestimmten Signalverarbeitungselemente einmal zwischen sukzessiven Samples, wobei das Verfahren in der Lage ist, Datensignal-Rückkopplung in dem Feld von Instrumentationseinheiten aufzulösen;
Anzeigen der Feldausgabe auf der Videoanzeige (10) in Echtzeit;
Speichern der Feldausgabe in dem Speicher (3); und
Bereitstellen der Feldausgabe an den Signalausgang des multifunktionalen Eingangs-/Ausgangssignalmoduls in Echtzeit.

30

35

9. Verfahren gemäß Anspruch 8, bei welchem ein Parameter des funktionalen Blocks spezifiziert werden kann.
10. Verfahren gemäß Anspruch 8, bei welchem das Feld zumindest eine Datensignal-Rückkopplungsschleife umfasst, in welcher ein Ausgang von einer der Instrumentationseinheiten verbunden ist, um einen Eingang von einer der Mehrzahl von Instrumentationseinheiten zu bilden, und ein ursprüngliches Ergebnis eines entsprechenden der ausführbaren Ereignisse in der Sequenz ausführbarer Ereignisse verwendet wird in einer wiederholten Ausführung eines früheren Ereignisses, um ein modifiziertes Ergebnis des entsprechenden einen der ausführbaren Ereignisse von einer vorherigen Abtastperiode zu erzeugen.

40

Revendications

45

50

55

1. Emulateur d'instrument scientifique comportant :

un ordinateur (1) incluant une mémoire (3) couplée à un processeur (2),
un module d'entrée/sortie de signal multifonctionnel (6) ayant au moins une entrée de signal analogique (7) ayant des moyens de numérisation (37) qui lui sont connectés et au moins une sortie de signal (8) et étant configurable par rapport à au moins une fréquence d'échantillonnage de ladite entrée de signal analogique (7), un affichage vidéo (10) couplé à un générateur d'affichage vidéo (50), ledit générateur d'affichage vidéo (50) étant couplé audit processeur (2),
un programme de bibliothèque mémorisé dans ladite mémoire (3), ledit programme de bibliothèque incluant une pluralité d'unités d'instrumentation chacune pouvant être affichées sur ledit affichage vidéo (10) et ayant chacune une fonction de traitement de signal prédéterminée,
un programme d'initialisation mémorisé dans ladite mémoire (3) et exécuté par ledit processeur (2) pour permettre à un opérateur d'interconnecter des unités d'instrumentation sélectionnées pour former un ensemble d'unités d'instrumentation interconnectées, dans lequel une fréquence d'échantillonnage dudit module d'entrée/sortie de signal multifonctionnel (6) est spécifiée, ledit programme d'initialisation permet une interconnexion à rétroaction de signal dans ledit ensemble, ledit ensemble ayant une sortie fournie à au moins l'un des éléments suivants : ledit affichage vidéo, ledit module d'entrée/sortie de signal multifonctionnel (6) et ladite mémoire (3), et

- un programme de compilateur graphique mémorisé dans ladite mémoire (3) et exécutable par ledit processeur (2) à la fin dudit programme d'initialisation pour créer un programme d'émulation d'instrument scientifique permettant d'exécuter une fonction de traitement de signal complète dudit ensemble d'unités d'instrumentation interconnectées sélectionnées, ledit programme de compilateur graphique étant capable de résoudre une
- 5 rétroaction de signal dudit ensemble et de former une séquence dans ledit programme d'émulation desdites fonctions de traitement de signal prédéterminées desdites unités d'instrument, qui doivent être chacune exécutées une fois entre des échantillons successifs d'un signal d'entrée obtenu en utilisant ledit module d'entrée/sortie de signal multifonctionnel, ledit programme d'émulation contenant des opérations configurées conformément à ladite fréquence d'échantillonnage spécifiée pour échantillonner ledit signal d'entrée afin de fournir
- 10 un flot de données à partir dudit module d'entrée/sortie de signal multifonctionnel (6), un sous-groupe d'émulation d'instrument couplé audit processeur (2), à ladite mémoire (3) et audit générateur d'affichage vidéo, où ledit sous-groupe d'émulation d'instrument est directement connecté audit module d'entrée/sortie de signal multifonctionnel, ledit sous-groupe d'émulation d'instrument ayant : un second processeur pour exécuter ledit programme d'émulation d'instrument scientifique pour réaliser en continu ladite fonction
- 15 de traitement de signal complète de telle sorte que chaque échantillon dudit flot de données d'échantillons successifs est traité pendant une seule durée d'échantillon pour produire une sortie d'ensemble dudit émulateur d'instrument scientifique avant qu'un échantillon successif dudit signal d'entrée ne soit obtenu par ledit module d'entrée/sortie de signal multifonctionnel (6),
- 20 dans lequel au moins l'une d'une pluralité d'opérations est effectuée, ladite pluralité d'opérations incluant chacune l'affichage de ladite sortie d'ensemble sur ledit affichage vidéo (10) en temps réel, la mémorisation de ladite sortie d'ensemble dans la mémoire (3), et l'envoi de ladite sortie d'ensemble à ladite sortie de signal dudit module d'entrée/sortie de signal multifonctionnel (6) en temps réel.
- 25 2. Emulateur d'instrument scientifique selon la revendication 1, dans lequel lesdites unités d'instrumentation sont chacune représentées par une icône correspondante qui peut être affichée sur ledit affichage vidéo pendant l'exécution dudit programme d'initialisation par lequel chaque unité d'instrumentation souhaitée peut être sélectionnée, localisée, et interconnectée avec d'autres unités d'instrumentation dudit ensemble.
- 30 3. Emulateur d'instrument scientifique selon la revendication 2, dans lequel ledit ensemble inclut une boucle de rétroaction de signal de données connectée entre une sortie de l'une desdites unités d'instrumentation et une entrée de l'une desdites unités d'instrumentation.
- 35 4. Emulateur d'instrument scientifique selon la revendication 1, dans lequel ledit module d'entrée/sortie de signal multifonctionnel inclut une sortie de signal supplémentaire à relier à un matériel externe et qui est disponible en tant que signal électrique en temps réel pouvant commander, ou déclencher, ledit matériel externe.
- 40 5. Emulateur d'instrument scientifique selon la revendication 4, dans lequel ledit module d'entrée/sortie de signal multifonctionnel a à la fois des sorties de signal analogique et numérique.
- 45 6. Emulateur d'instrument scientifique selon la revendication 4, dans lequel ladite sortie d'ensemble mémorisée dans cette mémoire est disponible pour une manipulation et/ou tabulation graphique ultérieure par ledit ordinateur.
- 50 7. Emulateur d'instrument scientifique selon la revendication 1, dans lequel ledit assemblage de processeur d'émulation d'instrument comporte en outre un générateur vidéo connecté audit second processeur pour fournir une représentation affichable de ladite sortie d'ensemble avant d'obtenir un échantillon suivant.
- 55 8. Procédé pour déterminer l'ordre d'exécution d'éléments de traitement de signal prédéfinis dans un ordinateur ayant une unité centrale de traitement (2) et une mémoire électronique (3) pour implémenter en temps réel une fonction mathématique/de traitement de signal d'un instrument émulé ayant au moins une entrée de signal échantillonnée d'une manière répétée pour former une sortie de signal de celui-ci, ledit procédé comportant les étapes consistant à :
- appliquer un programme d'initialisation mémorisé dans ladite mémoire (3) et exécuté par ladite unité centrale de traitement (2) pour permettre à un opérateur d'interconnecter des unités d'instrumentation sélectionnées, dans lequel une fréquence d'échantillonnage d'un module d'entrée/sortie de signal multifonctionnel (6) est spécifiée, ledit programme d'initialisation permet l'interconnexion à rétroaction de signal dans ledit ensemble, ledit ensemble ayant une sortie fournie à au moins l'un des éléments suivants : l'affichage vidéo (10), ledit

EP 0 573 503 B1

module d'entrée/sortie de signal multifonctionnel (6) et ladite mémoire (3), chacune desdites unités d'instrumentation ayant une fonction mathématique/de traitement de signal, créer un programme d'émulation d'instrument scientifique en utilisant un programme de compilateur graphique mémorisé dans ladite mémoire (3) et exécutable par ladite unité centrale de traitement (2) à la fin dudit programme d'initialisation, ledit programme d'émulation d'instrument scientifique réalisant ladite fonction de traitement de signal complète dudit ensemble d'unités d'instrumentation interconnectées sélectionnées, déterminer un ordre d'exécution de ladite fonction mathématique/de traitement de signal incluant les étapes consistant à :

représenter ladite fonction sous la forme d'une séquence d'étapes mathématiques élémentaires pouvant être à leur tour représentées en utilisant des éléments de traitement de signal prédéfinis, et organiser les éléments de traitement de signal prédéfinis en ladite séquence pour une exécution séquentielle pendant la durée entre des échantillons successifs et commencer par ladite au moins une entrée de signal échantillonnée, dans lequel ladite étape de détermination forme ainsi un ensemble d'unités d'instrumentation ayant ladite fonction mathématique/de traitement de signal de circuit complète, réaliser ladite fonction de traitement de signal complète de telle sorte que chaque échantillon dudit flot de données contiguës est traité pendant une seule durée d'échantillon pour produire une sortie d'ensemble de ladite fonction mathématique/de traitement de signal complète avant d'obtenir un échantillon successif dudit signal d'entrée par ledit module d'entrée/sortie de signal multifonctionnel (6), représenter ladite fonction mathématique/de traitement de signal de circuit complète sous la forme d'une séquence d'événements représentable sous la forme d'événements d'éléments de traitement de signal prédéfinis, organiser lesdits événements d'éléments de traitement de signal prédéfinis en ladite séquence pour une exécution séquentielle en commençant par ladite au moins une entrée de signal d'ensemble échantillonnée, de manière à ce qu'une durée de calcul déterminée des éléments de traitement de signal prédéfinis soit garantie en exécutant chacun desdits éléments de traitement de signal prédéfinis une fois entre des échantillons successifs, ainsi ledit procédé est capable de résoudre une rétroaction de signal de données dudit ensemble d'unités d'instrumentation, afficher ladite sortie d'ensemble sur ledit affichage vidéo (10) en temps réel, mémoriser ladite sortie d'ensemble dans la mémoire (3), et fournir ladite sortie d'ensemble à ladite sortie de signal dudit module d'entrée/sortie de signal multifonctionnel en temps réel.

9. Procédé selon la revendication 8, dans lequel un paramètre dudit bloc fonctionnel peut être spécifié.

10. Procédé selon la revendication 8, dans lequel ledit ensemble inclut au moins une boucle de rétroaction de signal de données dans laquelle une sortie de l'une desdites unités d'instrumentation est connectée pour former une entrée d'une unité de ladite pluralité d'unités d'instrumentation, et un résultat initial d'un événement correspondant parmi lesdits événements exécutables de ladite séquence événements exécutables est utilisé dans une exécution répétée d'un événement antérieur pour générer un résultat modifié dudit événement correspondant parmi lesdits événements exécutables à partir d'une précédente période d'échantillon.

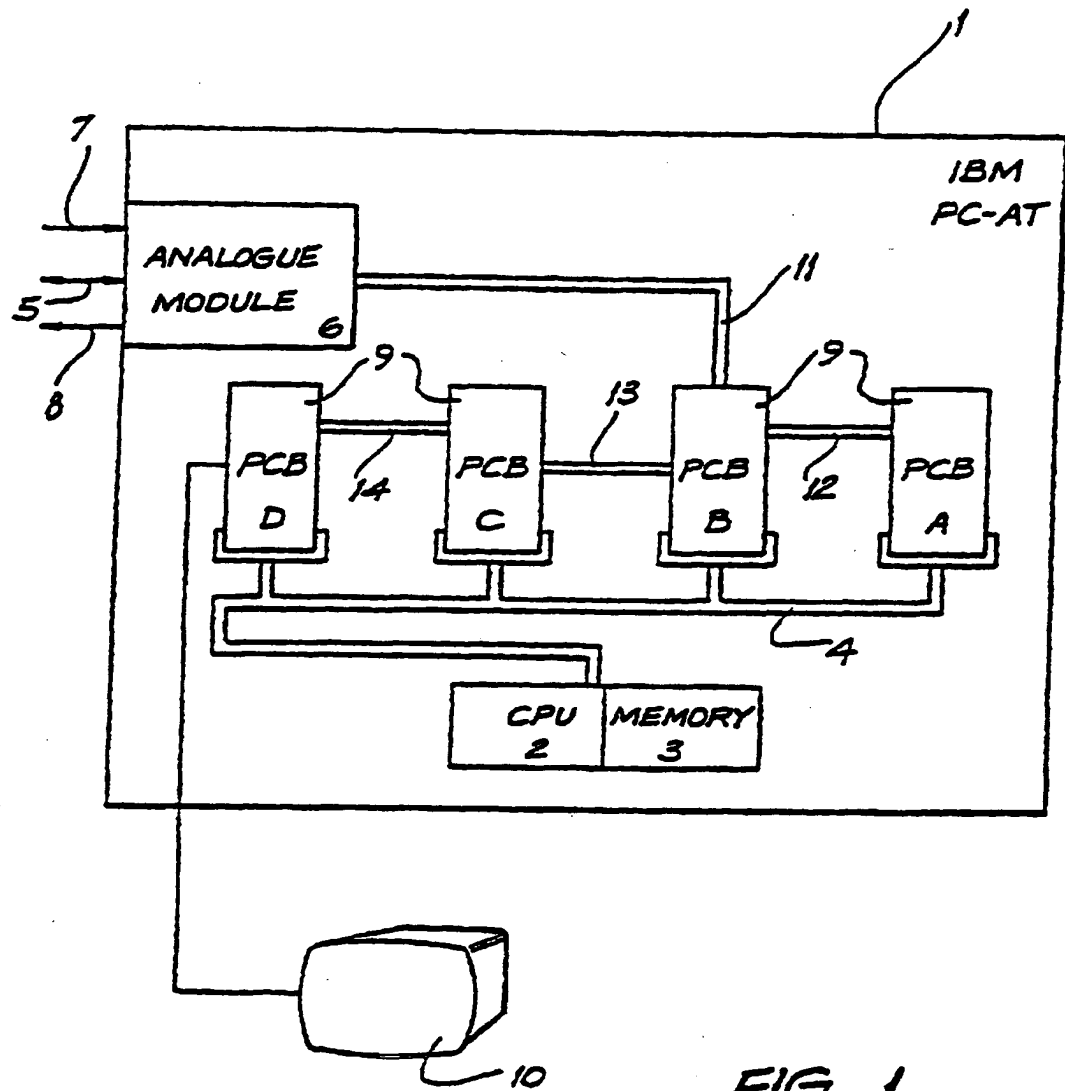


FIG. 1

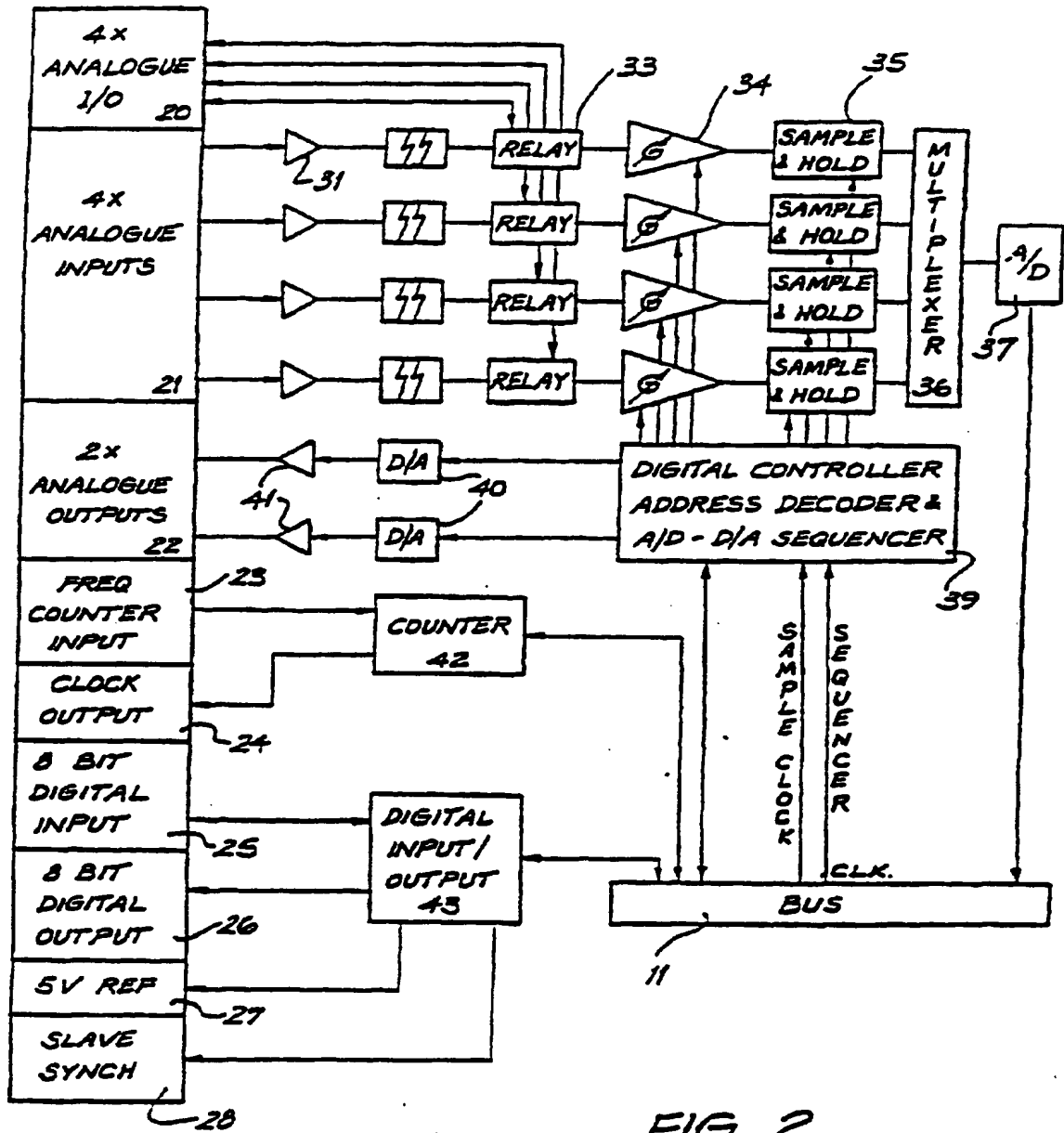


FIG. 2

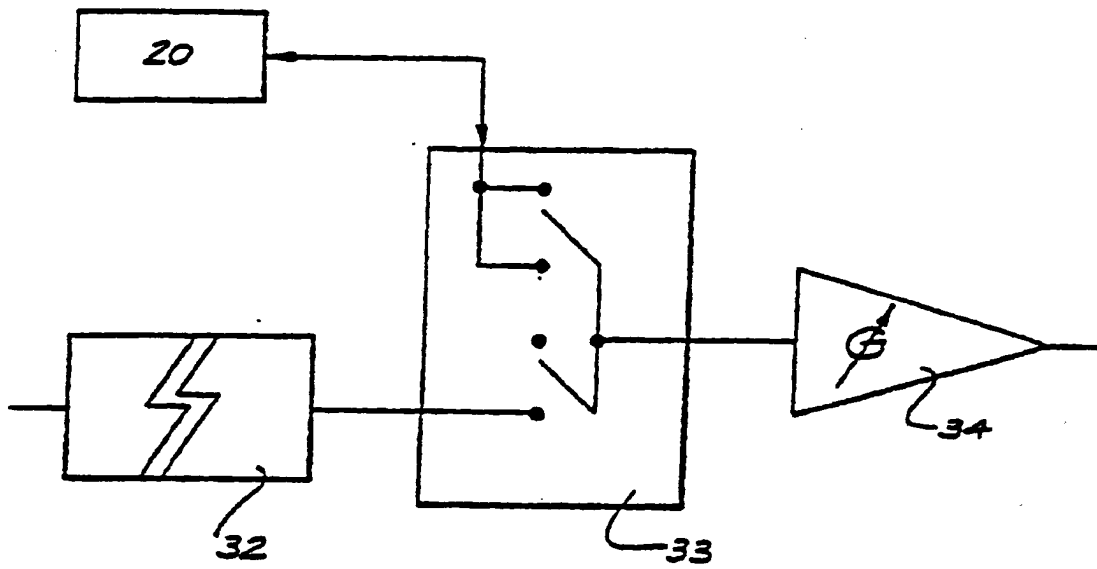
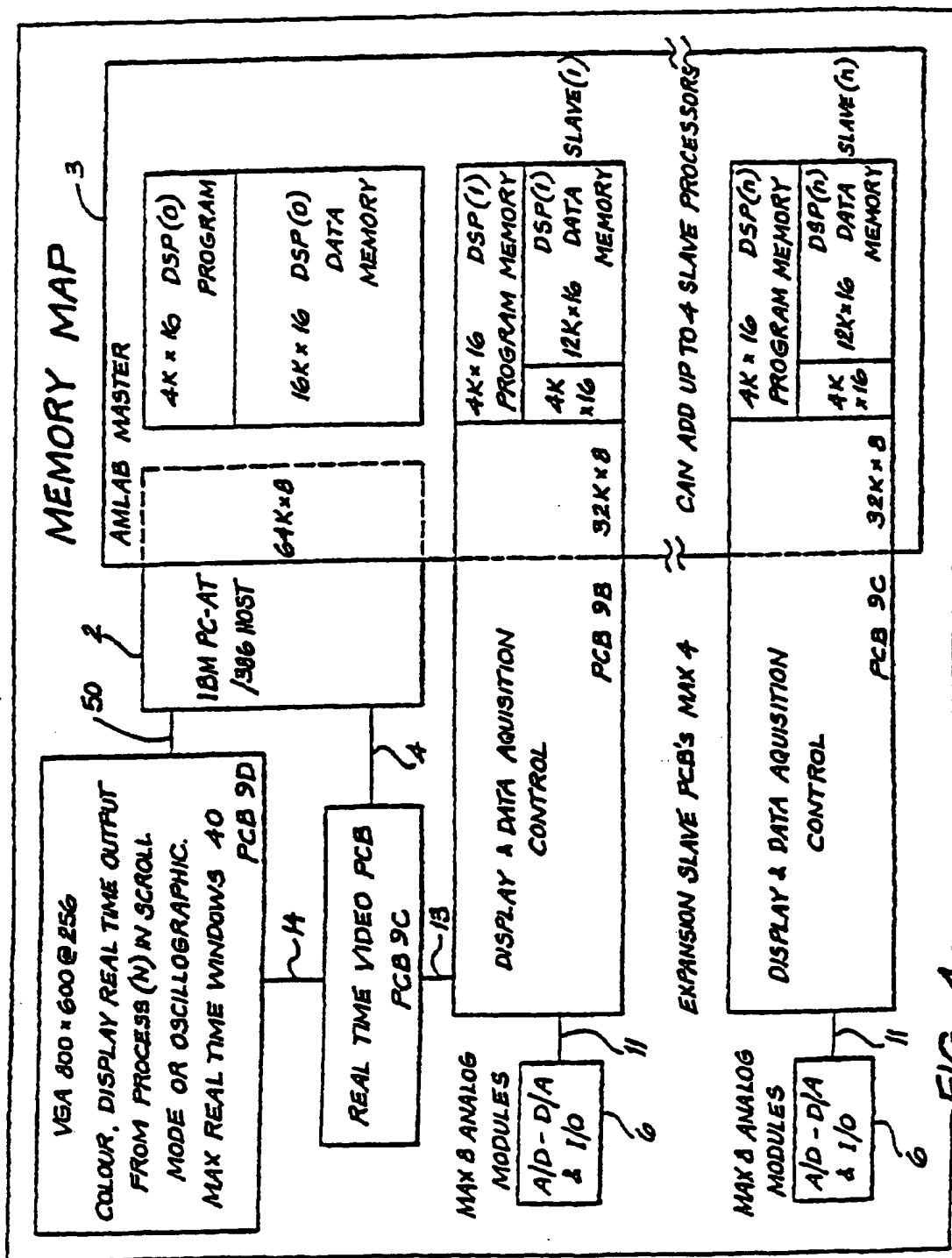


FIG. 3



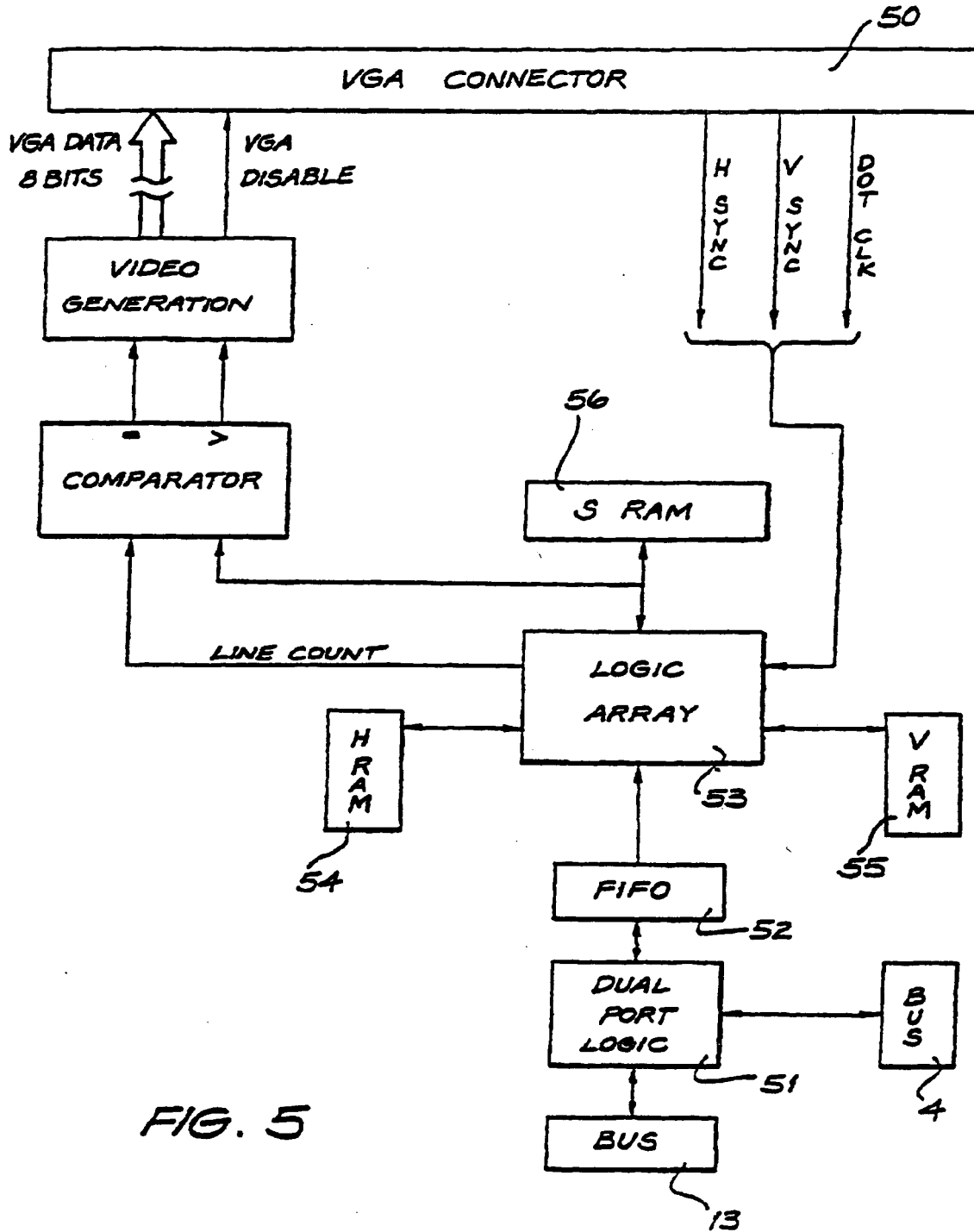


FIG. 5

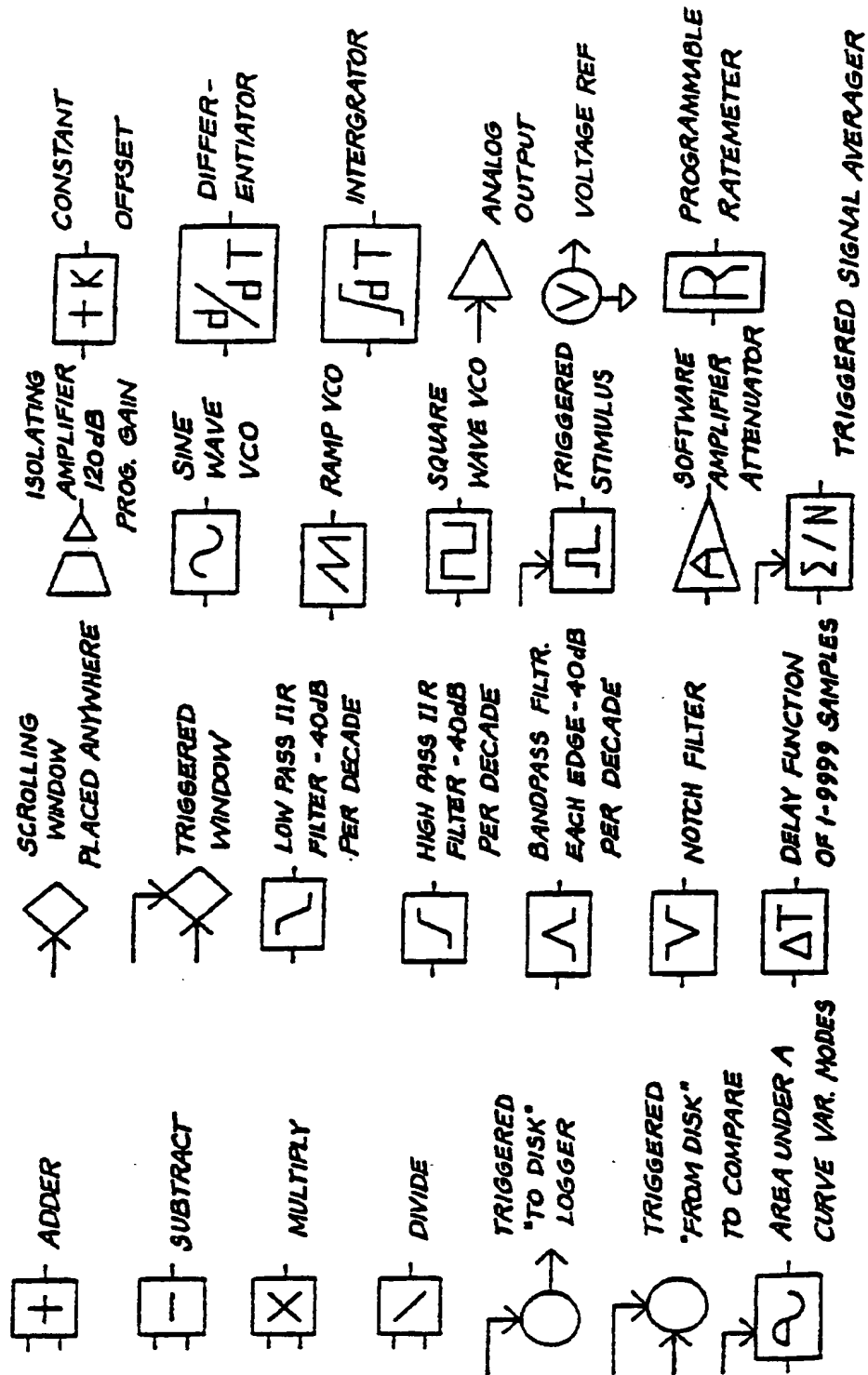


FIG. 6

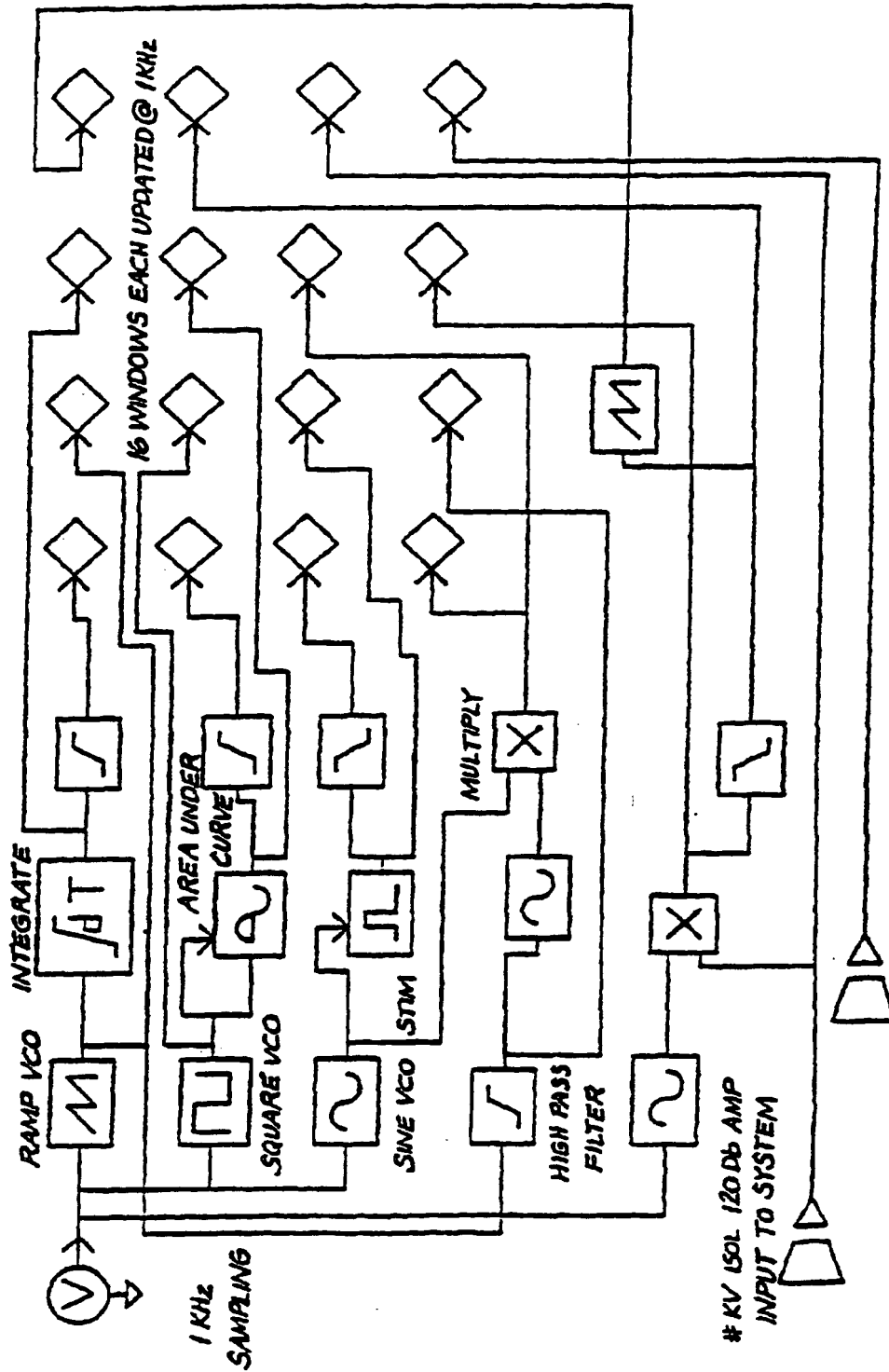


FIG. 7

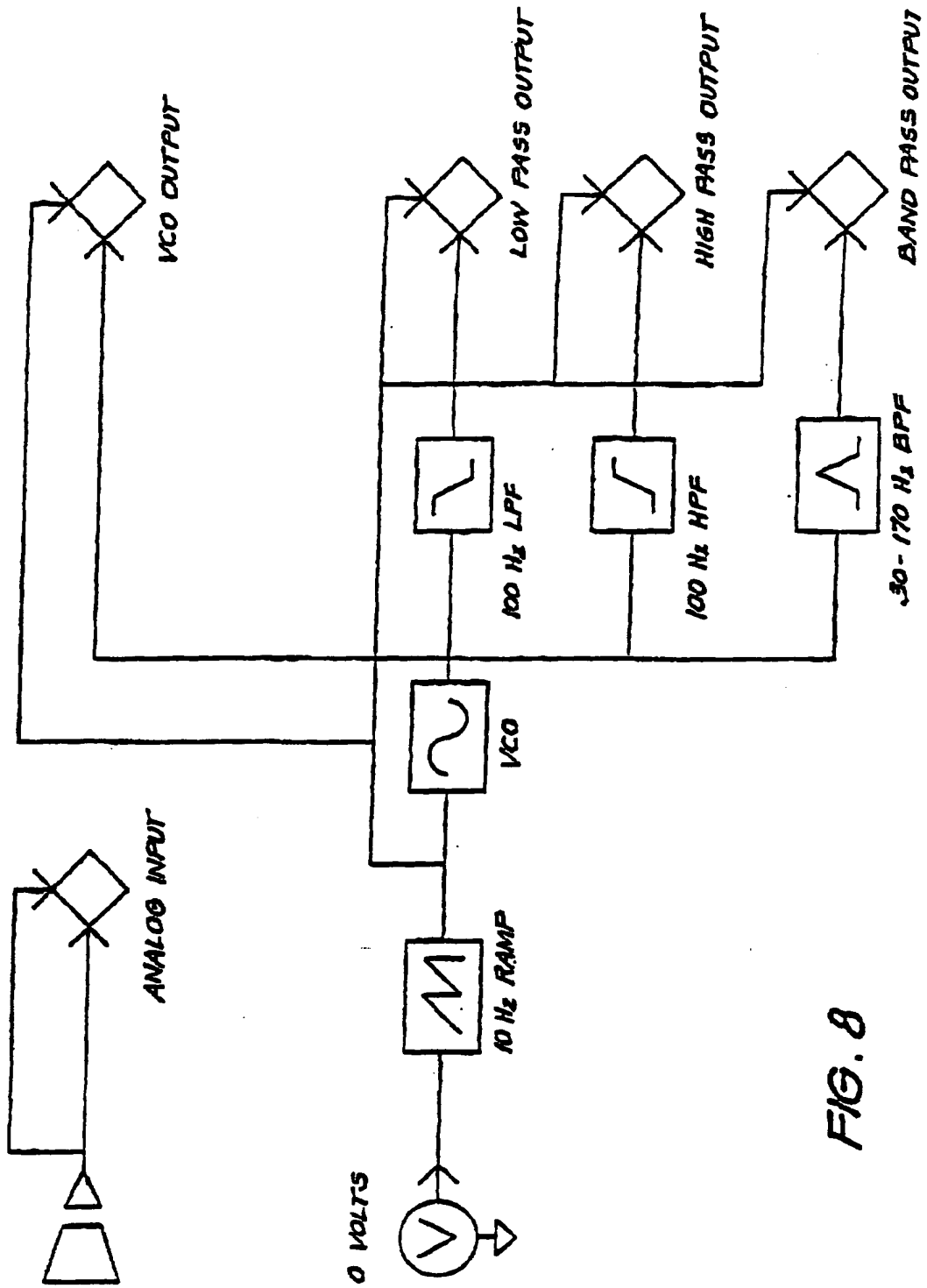


FIG. 8

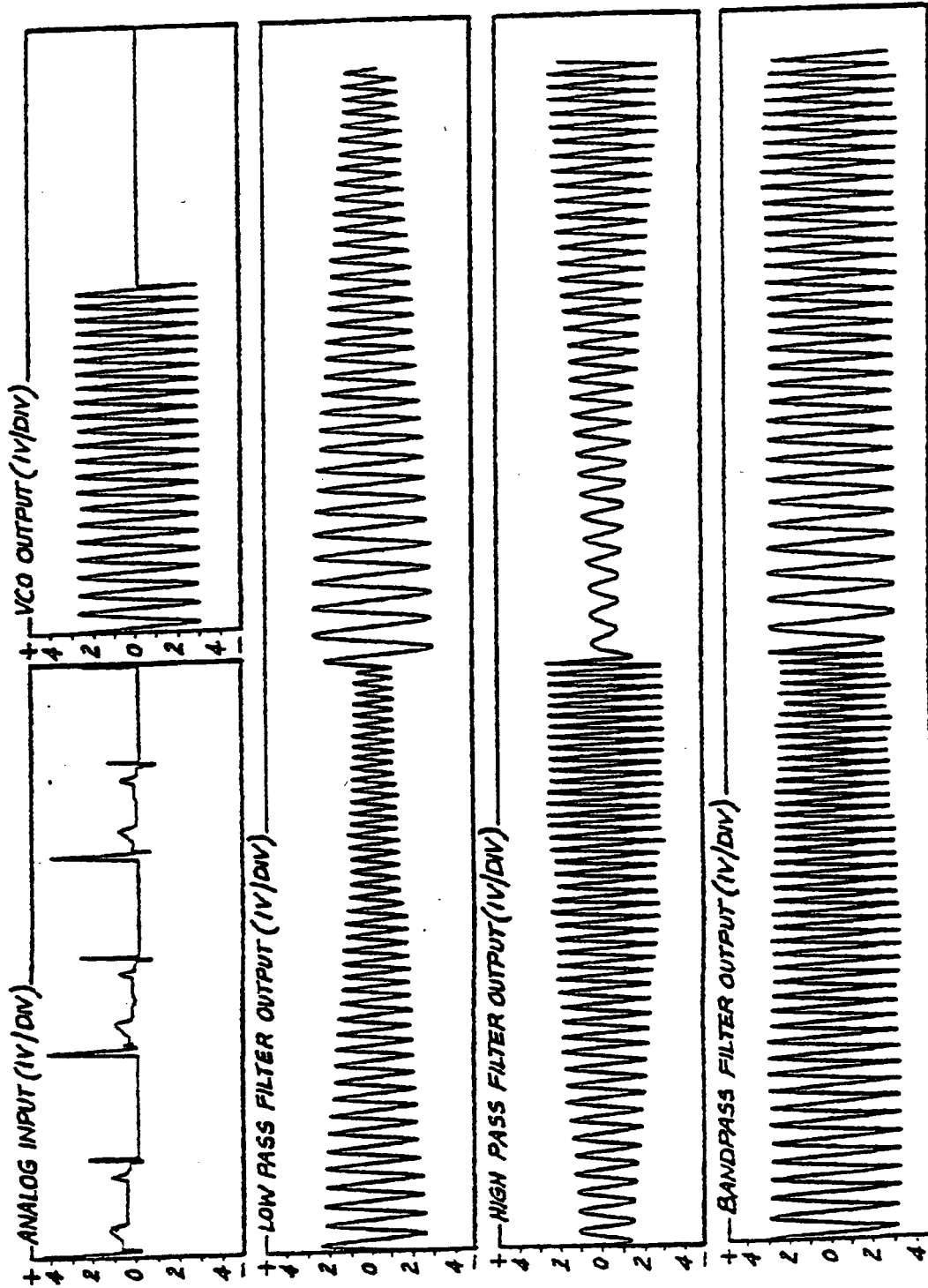


FIG. 9

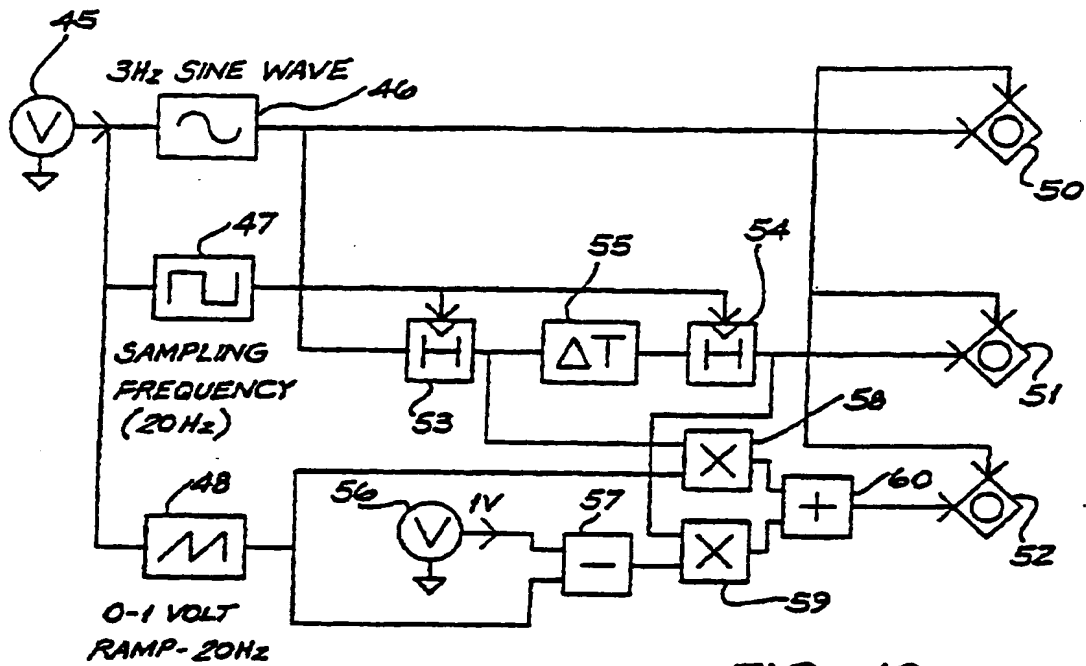


FIG. 10

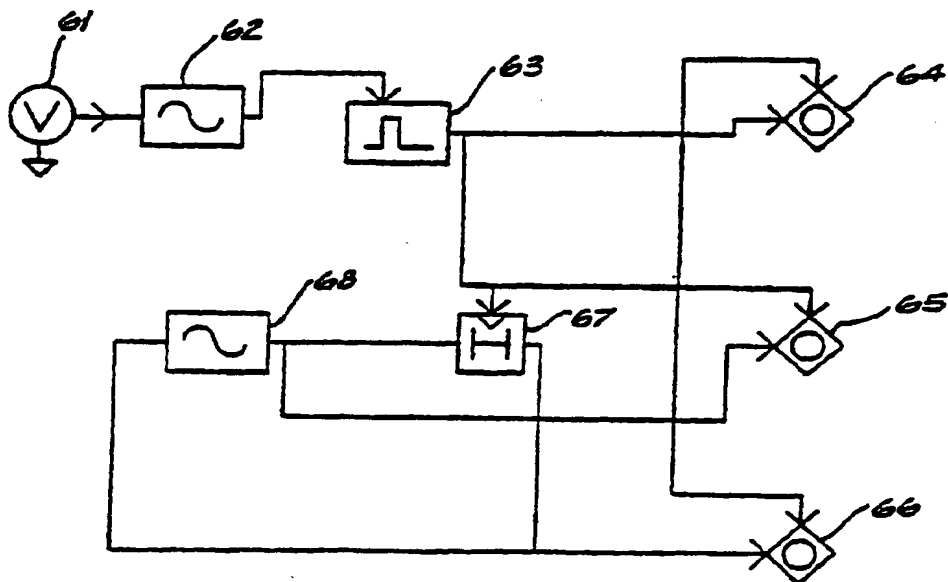


FIG. 12

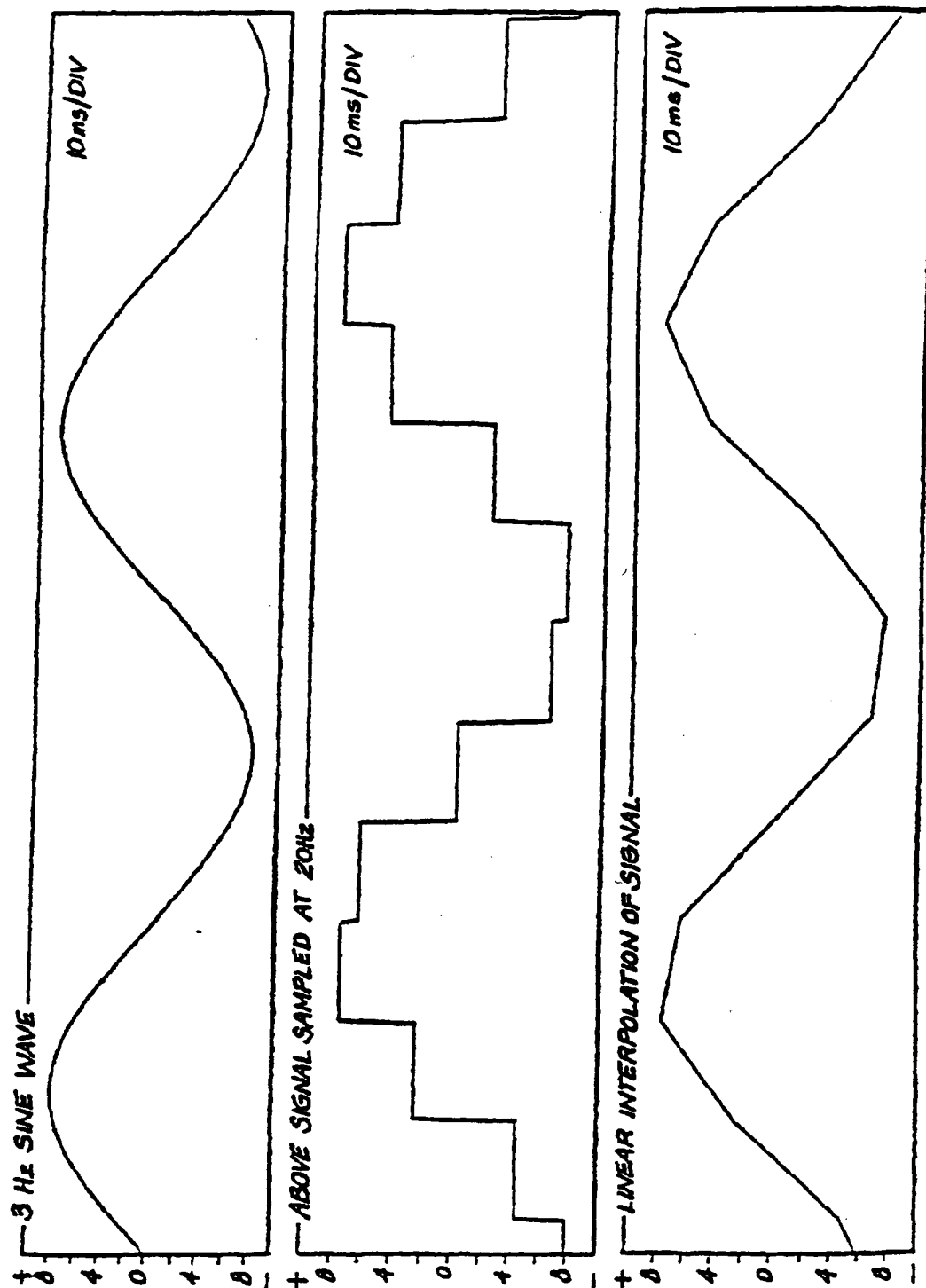


FIG. 11

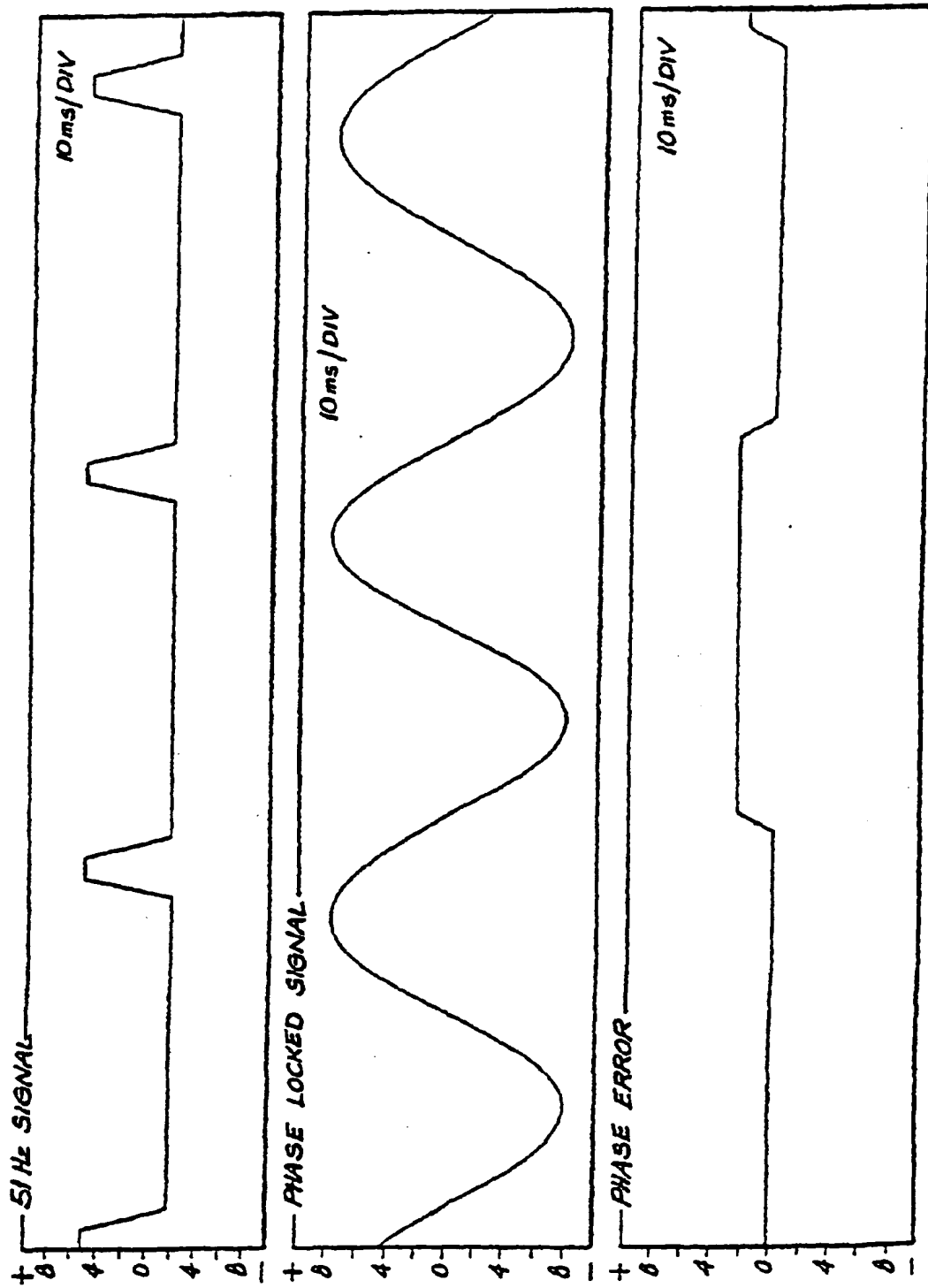


FIG. 13

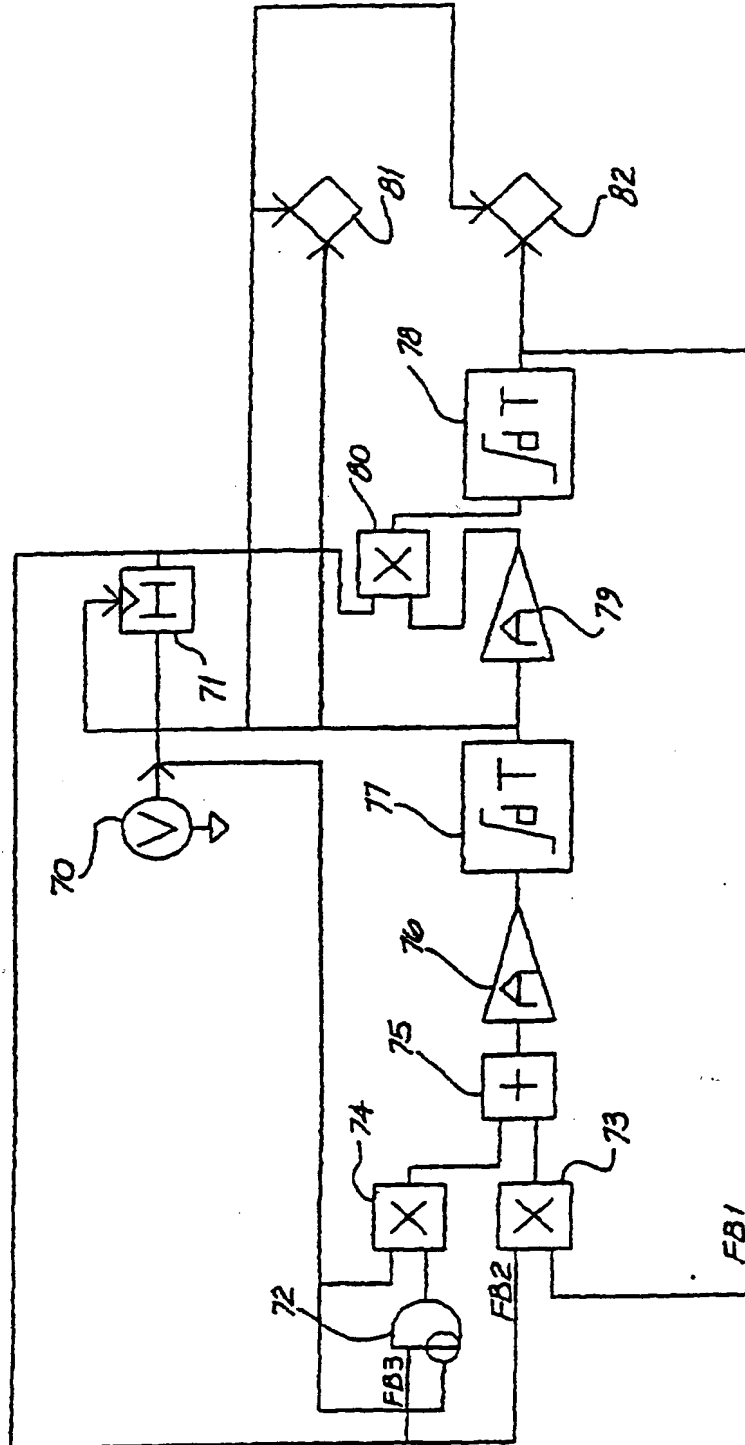


FIG. 14

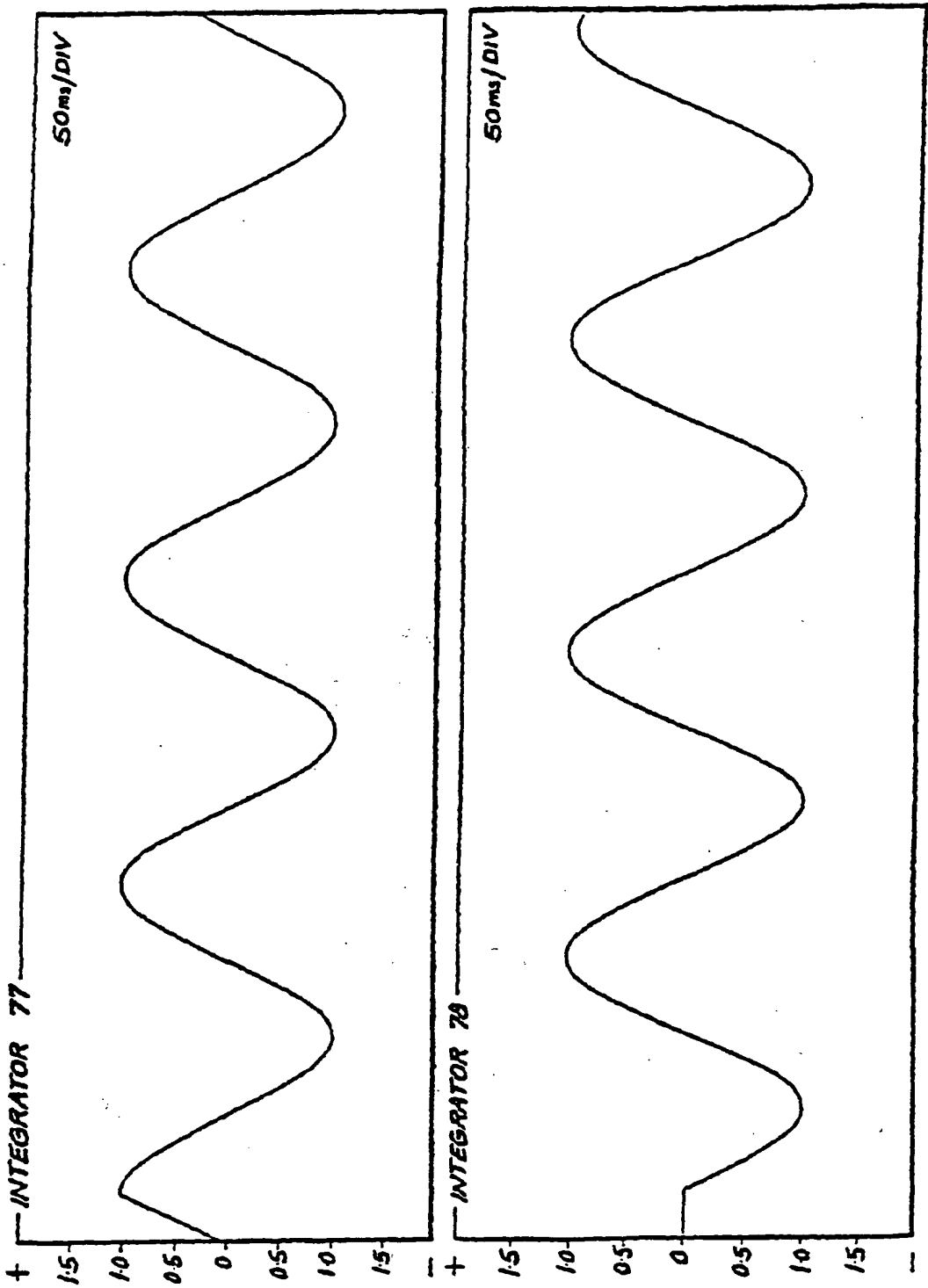


FIG. 15

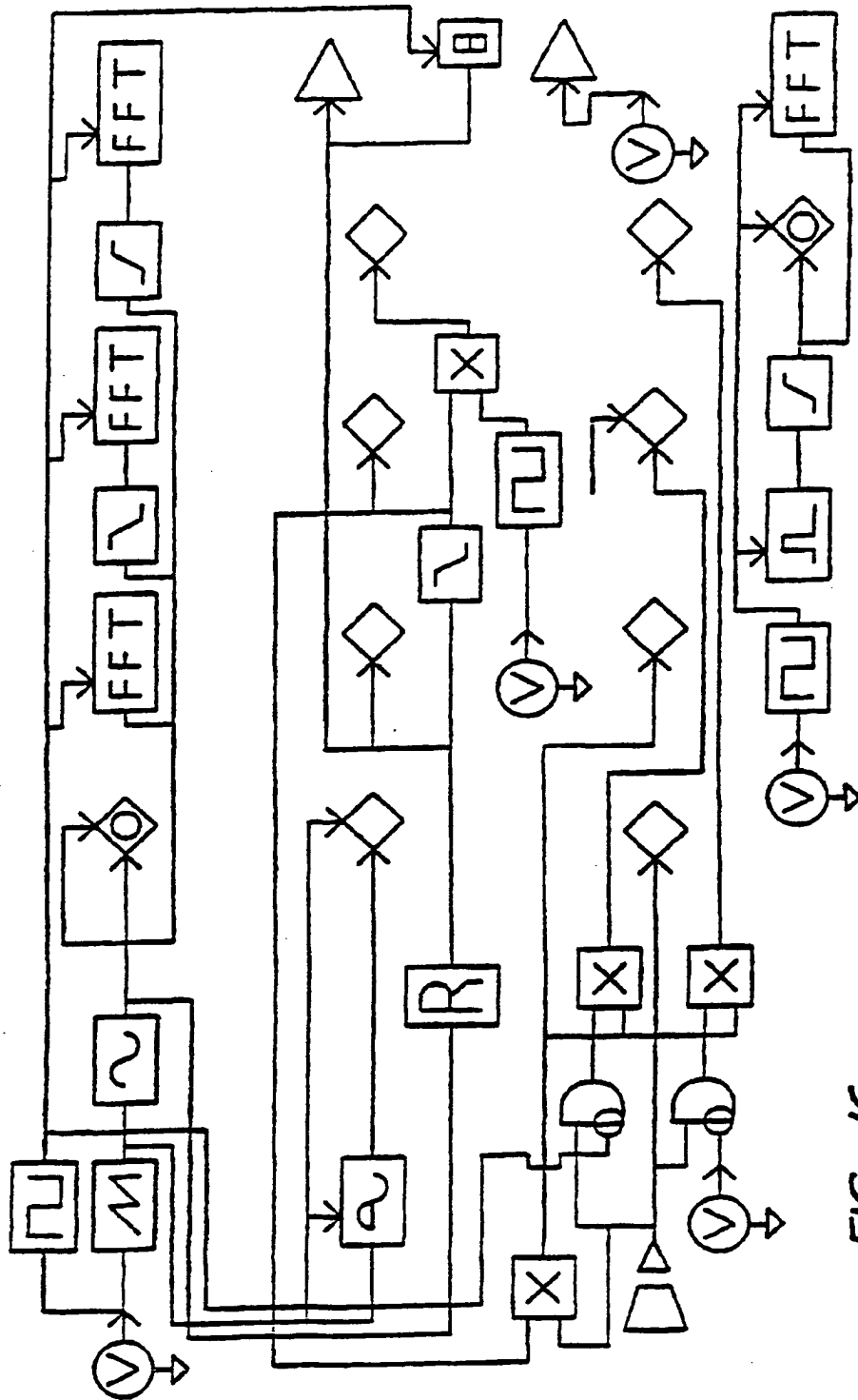


FIG. 16

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ **BLACK BORDERS**
- ☐ **IMAGE CUT OFF AT TOP, BOTTOM OR SIDES**
- ☐ **FADED TEXT OR DRAWING**
- ☐ **BLURRED OR ILLEGIBLE TEXT OR DRAWING**
- ☐ **SKEWED/SLANTED IMAGES**
- ☐ **COLOR OR BLACK AND WHITE PHOTOGRAPHS**
- ☐ **GRAY SCALE DOCUMENTS**
- ☐ **LINES OR MARKS ON ORIGINAL DOCUMENT**
- ☐ **REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY**
- ☐ **OTHER:** _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.

THIS PAGE BLANK #1007